CORE
Metadata, citation and similar papers at core.ac.uk
Provided by IMT E-Theses

IMT Institute for Advanced Studies, Lucca

Lucca, Italy

Novel Routing Paradigms for Wireless Mesh Networks

PhD Program in Computer Science and Engineering

XXIII Cycle

By

Maddalena Nurchis

2011

The dissertation of Maddalena Nurchis is approved.

Program Coordinator: Prof. Rocco De Nicola, IMT Lucca, Italy

Supervisor: Prof. Luciano Lenzini, University of Pisa, Italy

Supervisor: Dr. Marco Conti, IIT - CNR, Pisa, Italy

Tutor: Dr. Leonardo Badia, University of Padova, Italy

The dissertation of Maddalena Nurchis has been reviewed by:

Vasilios A. Siris, University of Athens; ICS-FORTH, Crete, Greece

Luciano Bononi, University of Bologna, Italy

IMT Institute for Advanced Studies, Lucca

2011

To my family

Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank my supervisors, Prof. Lenzini and Dr. Conti, for their precious guide during the research activity of my Ph.D. They have encouraged and supported me in investigating interesting topics and designing innovative solutions. Their contribution has been fundamental to achieve these results.

I am very grateful to Raffaele Bruno, who significantly contributed to this thesis, not only with his work, but also with his advises and suggestions, helping me in improving my knowledge and skills.

A special thank to my family and all my friends, who have supported and helped me during my Ph.D. activity.

Vita

December 26, 1981	Born, Nuoro, Italy
2006	Erasmus Student University of Denmark, Odense, Denmark
2007	Master Degree in Computer Engineering University of Pisa, Italy
since 2008	PhD Student IMT Lucca Institute for Advanced Studies, Italy
July, 2008	Student International School "Algorithms: Science and Engineering". Lipari School, Lipari, Siciliy.
July, 2009	Student Future Internet Summer School. University of Bremen, Germany.
since January 2010	Research Associate National Research Council (CNR), Pisa
September - December 2010	Visiting Student Prasant Mohapatra's Network Research Group Department of Computer Science, Davis, CA

Publications

1. R. Bruno and M. Nurchis, "Survey on diversity-based routing in wireless mesh networks: challenges and solutions ", in *Computer Communications*, vol. 33, no. 3, pp. 269 – 282, February 2010.
2. R. Bruno, M. Conti and M. Nurchis, "MaxOPP: A Novel Opportunistic Routing for Wireless Mesh Networks", *Proc. of IEEE ISCC 2010*, June 2010.
3. R. Bruno, M. Conti and M. Nurchis, "Opportunistic packet scheduling and routing in wireless mesh networks", *Proc. of IFIP Wireless Days'10*, October 2010.
4. M. Nurchis, R. Bruno, M. Conti and L. Lenzini, "A Self-Adaptive Routing Paradigm for Wireless Mesh Networks Based on Reinforcement Learning", *Proc. of ACM MSWiM 2011*, November 2011.
5. R. Bruno, M. Conti and M. Nurchis, "RELADO: RELiable and ADaptive Opportunistic Routing Protocol for Wireless Mesh Networks", *to appear in IGI International Journal of Adaptive, Resilient and Autonomic Systems (IJARAS), IGI Global*, 2011.

Abstract

The increasing desire of ubiquitous Internet access has recently promoted the deployment of wireless multi-hop networks in several application domains. Wireless Mesh Networks (WMNs) provide significant benefits over existing wireless multi-hop networking paradigms, offering a suitable solution for a wide range of application scenarios, spanning from public safety communications to community-based networks and metro scale municipal networks.

Routing design is crucial to guarantee robust communication through the mesh backbone. Traditional unicast routing has shown to be ineffective when dealing with highly variable wireless channels. One of the most critical aspects is the *wireless diversity*, intended as the reception of a packet at multiple forwarders, causing collisions and interference due to the broadcast nature of the wireless medium. A set of innovative routing approaches has recently been proposed as a valuable alternative to classical routing, thanks to their ability to deal with the wireless diversity as an opportunity rather than a shortcoming.

The primary goal of this thesis is to deeply investigate wireless diversity-based routing in WMNs, proposing novel solutions able to significantly improve WMN performance. We extensively describe the main features of this strategy and provide a classification of the most representative solutions in literature, discussing their most relevant characteristics, advantages and disadvantages. Then, we focus on one of the most promising categories: Opportunistic Routing (OR). It exploits the multiple packet recipients offered by the wireless transmission to incrementally build a path, selecting the

best next hop only after packet reception. Then, we propose a novel opportunistic routing algorithm, able to select at each hop the forwarders that maximize the throughput gain. In contrast to the common opportunistic approach, the proposed algorithm avoids any form of a priori constraint on route selection, fully leveraging all the transmission opportunities encountered during path construction. To improve its efficiency in multi-flow environments, we extend its routing strategy with an opportunistic packet scheduling algorithm and a prioritized channel access scheme, so as to facilitate the transmission of the packets that are traversing the paths providing higher performance gains.

To ensure high performance in all the typical WMN application scenarios, we need to consider that in these environments channel quality may significantly vary in time and space, requiring a high degree of flexibility in the path construction process. Most of the existing solutions perform local decisions (i.e. hop-by-hop) based on end-to-end principles. In contrast, we propose a novel routing algorithm that combines end-to-end with localized data, so as to adapt routing decisions to channel conditions at the time of packet transmission. This ensures higher reliability even in the most challenging application scenarios.

The key factors determining opportunistic improvements are not clear yet, making hard to identify the conditions under which this paradigm outperforms classical unicast routing. Hence, we propose a novel routing architecture that relies on a configurable machine learning-based agent to properly select, at each node, the most suitable routing algorithm within a set of available solutions, according to network conditions and traffic characteristics. This solution represents a further step towards the definition of a wireless diversity-based routing paradigm able to ensure high performance in all WMN application scenarios.

Chapter 1

Introduction

In the last decade, wireless networks have rapidly revolutionized our lifes, totally changing our way of communication. Recent advances in wireless technology together with standardization efforts have permitted the wide-spread deployment of wireless networks to satisfy the increasing desire of being connected *anytime* and *anywhere*. Clearly, the high penetration of wireless networks into every aspect of our daily life has required the adoption of networking paradigms able to provide flexible, low-cost and easy-to-deploy connectivity.

A crucial role is played by wireless multi-hop networks, which are composed by devices that cooperatively relay traffic between nodes that can not directly communicate. Hence, multi-hop network paths can be established between any pair of nodes without relying on a pre-existing network infrastructure or dedicated network devices (e.g. routers, switches, servers) (CG04b). This distributed networking paradigm has been originally proposed for military networks. Recently, the advent of new mobile devices (e.g. smartphones), and the growing interest of the community in accessing connectivity services have promoted its utilization for a variety of innovative application domains, ranging from sensor networks to vehicular networks and mesh networks (CG04a). In particular, wireless mesh networks are one of the most attracting applications of this networking paradigm, due to their inherent capability to reduce

cost and complexity of network configuration and maintenance. Indeed, they are static ad hoc networks consisting of dedicated nodes that form a multi-hop wireless backbone used to share a limited number of fixed Internet connections with a potentially large number of static or nomadic users (AWW05).

Nowadays, in order to provide Internet access, cellular networks are commonly used in scenarios such as metropolitan areas, whereas IEEE 802.11-based WLANs are typically deployed to provide home, community or enterprise networking. The former can efficiently cover a wide area, but achieve low data rates and require the deployment of a complex network infrastructure, which is expensive and might not be feasible in challenging environments. The latter permit to provide Internet access at higher data rates and lower costs, although they are not suitable for large environments, where the installation of multiple access points may be expensive and hard to realize in many cases.

The wireless mesh architecture combines the robustness of the wireless infrastructure (i.e. mesh backbone) with the flexibility of an incremental deployment. The former is guaranteed by a set of static routers not subject to strict power constraints and responsible for routing packets across the network. The latter is allowed by the self-configuration and self-maintenance capabilities, which permit to deploy a mesh network also in challenging environments and to gradually extend it as needed. Thanks to the multi-hop communication paradigm, coverage can be easily extended without requiring high transmission power, and even a small set of routers connected to the Internet is sufficient to provide Internet access to a large set of users. For these reasons, WMNs offer several advantages over existing wireless networking paradigms, such as low cost, easy of incremental deployment and resilience against node/link failures. Hence, WMNs are expected to overcome limitations of existing infrastructure-based networking paradigms (e.g. cellular, Wi-Fi), providing a suitable solution for a wide range of application scenarios that can not be directly supported by other existing networks (AWW05).

The peculiarities of WMNs have stimulated a large body of research activities orientated to the re-design of algorithms and proto-

cols at all layers. Routing is fundamental in order to guarantee robust and low-overhead communications through the mesh backbone. The most intuitive approach for routing design is the application of the routing paradigm originally designed for the wired domain, i.e. the selection of the minimum-cost path between a source and a destination (JHM07; CJ03; PBRD03). This choice implies a point-to-point link abstraction (DADSC04) and relies on the assumption that link-layer retransmissions can effectively deal with packet losses. Although this approach has been the first attempt to make wireless mesh networks a reality, it has recently revealed its limitations when dealing with unreliability and unpredictability of wireless transmissions (BM05; RSMQ09; CRSK06; CJKK07; RSBA07).

Motivated by the above considerations, the research community has recently started to investigate radically new routing paradigms, able to turn the challenging peculiarities of wireless transmissions into an opportunity to improve network performance (BM05; RSMQ09; CJKK07; ZKR07; ZKR08; WLL08; YYW+05; KRH+08; DFGV07). Opportunistic routing is emerging as one of the most promising approaches to face performance degradation caused by lossy links and unpredictable channel conditions. Traditional unicast routing selects a sequence of nodes that each packet has to traverse to flow from the source to the destination. This choice usually imposes a critical trade-off: the distance of traversed links should be short enough to guarantee high delivery rate and long enough to ensure reasonable packet progress. Moreover, several retransmissions are required in presence of high packet loss rate, which is a common characteristic of wireless links (ABB+04; CRSK06). In contrast, opportunistic routing defers the selection of the next hop after packet reception, thus it can take advantage of the multiple transmission opportunities generated by the broadcast nature of the wireless medium. In fact, each packet may be received by multiple nodes, all experimenting different channel conditions. This novel strategy creates multiple paths for each endpoint pair by opportunistically exploiting long transmissions whenever possible, and relying on short hops in the other cases, ensuring high packet delivery rate even in presence of lossy

links (BM05). Reliability is provided through path redundancy, thus this routing strategy is mainly intended for bulk data transfers of long-lived flows, where in-order delivery is not the main concern. In contrast, the use of the scheme for real-time applications is not straightforward, since they impose strict timing and packet ordering constraints. Experimental results (BM05; RSMQ06; CJKK07) have demonstrated the high potentiality of this novel routing approach as well as its limitations when dealing with wireless links peculiarities, motivating the work presented in this thesis.

1.1 Thesis contribution

This thesis provides a significant contribution to the literature on diversity-based routing. The first contribution is an extensive overview of routing schemes for wireless mesh networks that exploit the reception of the same packet at multiple forwarders to improve network performance. In order to clarify the capability of this paradigm to provide performance gains, we build a classification of the main approaches, using it as a roadmap to analyze the design challenges that diversity-based routing needs to address. Then, we describe the features, advantages and disadvantages of the most representative solutions proposed in the literature and discuss the open issues in this research area.

Due to its potentiality, we focus on the opportunistic routing strategy and identify the main limitations of existing solutions. In order to maximize throughput performance, we propose MaxOPP, a novel opportunistic routing algorithm able to select the best forwarder(s) for each packet at run-time, rather than pre-computing a list of potential forwarders or imposing guard times before forwarders' transmissions, as performed in most of the existing solutions. This flexibility permits to achieve throughput gain compared to traditional shortest path routing, due to the ability to leverage all the opportunities encountered during packet forwarding. In order to improve efficiency in multi-flow scenarios, we propose an extension of MaxOPP algorithm, named PacketOPP, that combines opportunistic routing with opportunistic packet scheduling and prioritized

channel access, leading to higher throughput than existing opportunistic routing protocols.

The wide range and high variability of wireless mesh application scenarios entail improved flexibility and adaptability capabilities when designing routing solutions for wireless mesh networks. The multiple links traversed by a packet may present different loss rates, requiring to adapt the routing strategy along the whole path to ensure packet progress regardless of the channel characteristics. Indeed, the high variability of link quality may cause topology and routes instability similarly as mobility in Mobile Ad hoc Networks (aka MANETs). In addition, particular network conditions may exacerbate this instability, as may happen not only in challenged environments, such as emergency situations, but also in unplanned deployments of community-based networks. A routing algorithm designed for WMNs should be able to adapt the routing decision for each packet and at each hop according to the channel condition observed during packet transmission. Hence, we extend the basic opportunistic scheme mentioned above (MaxOPP) and we propose RELADO, which uses localized context (e.g. variability of link qualities in the neighborhood of the receivers) to adjust transmission redundancy and increase protocol reliability.

Despite the great deal of effort in investigating opportunistic routing ability to improve network performance, additional work is necessary to identify the fundamental conditions under which this routing paradigm outperforms traditional unicast schemes. Providing a response to the above question seems very difficult due to the large number of factors that can impact network performance of such heterogeneous scenarios, hence we propose a novel routing architecture that allows each node to efficiently select the most suitable routing protocol according to traffic pattern and channel conditions. It relies on a configurable machine learning-based agent to identify the most appropriate routing strategy among a set of available schemes so as to fit the requirements of each application scenario. The proper selection of the most suitable routing strategy is essential to achieve high performance in all application scenarios.

1.2 Thesis Structure

The extensive survey on wireless diversity-based routing protocols is provided in Chapter 2, introduced by the description and discussion of WMN architecture and benefits. Chapter 3 describes in details the two novel opportunistic routing algorithms proposed to maximize throughput gain: MaxOPP (3.2.1) and PacketOPP (3.3.1). Chapter 4 presents RELADO, the reliable and adaptive extension of our proposed basic opportunistic scheme. In Chapter 5 the machine learning-based hybrid protocol is extensively described and evaluated. Finally, in Chapter 6 we draw final conclusions and discuss open issues and ideas for future works.

Chapter 2

Wireless diversity-based routing

In this chapter, we provide the fundamental concepts behind the research activity presented in this thesis. Hence, we firstly describe the WMN architecture and discuss the most representative benefits of this promising technology. Then, we present some of the most common application scenarios. Finally, we provide an extensive overview of some of the most interesting routing approaches for WMNs that explicitly take advantage of wireless peculiarities. As mentioned above, this family of routing solutions has recently been proposed as an alternative to the classical unicast shortest path routing, and has demonstrated its potentiality to significantly improve network performance.

2.1 Wireless Mesh Networks

A typical WMN architecture is depicted in figure 1. A set of static Mesh Routers automatically forms and maintains a wireless backbone, which constitutes the infrastructure responsible for guaranteeing multihop communications within the network. Any static or mobile user (also called Mesh Client) can easily connect to the backbone by establishing a communication link with a nearby mesh router through any radio tech-

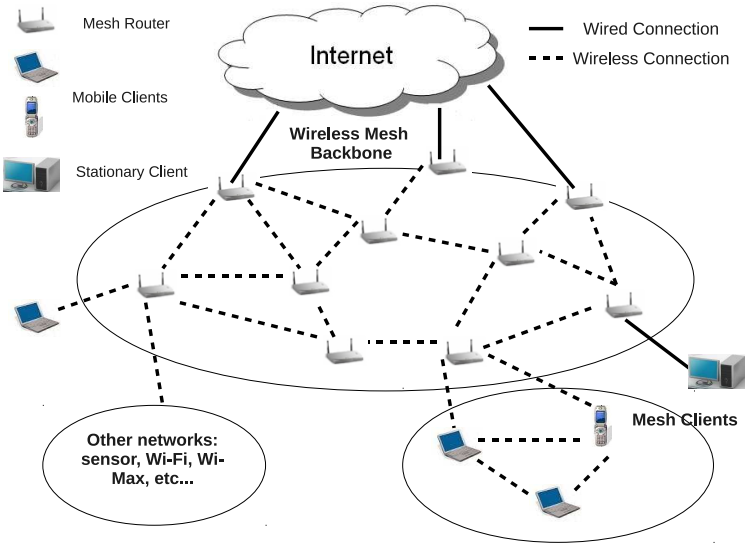


Figure 1: A typical Wireless Mesh Network architecture.

nology available at both nodes (e.g. Ethernet, 802.11, etc.). Then, each mesh client uses the multi-hop wireless backbone to connect with other mesh clients. In some cases, mesh clients can also form peer-to-peer networks among themselves and perform routing and configuration functionalities as mesh routers, becoming an active part of the mesh backbone rather than simply connecting to it. In addition, a few mesh routers, called Mesh Gateways, are also connected to the Internet, enabling the mesh backbone to easily extend the coverage provided by a limited number of gateways to all the mesh clients (AWW05). In addition to the Internet, mesh routers can act as gateways or bridges to many other external networks, such as cellular or sensor networks. The integration of the WMN with various existing networks provides users with the potentiality to access services in other networks.

The distinctive feature of this innovative networking architecture is the wireless static infrastructure, which provides significant advantages

over other networking paradigms. First, its static and dedicated nature ensures robust multi-hop communications regardless of the variable contribution of end-users, in contrast to the pure ad-hoc paradigm, where connectivity and reliability depends on users' connection and mobility. Second, the dedicated infrastructure allows the load on end-user devices to be significantly lower than that on mesh routers, thus decreasing the requirements on those devices and consequently their price and power consumption. Third, the network can easily be extended by incrementally adding mesh routers as needed, increasing reliability and connectivity. Moreover, a wired connection is required only for a restricted subset of mesh routers, thus it represents a flexible, cost-effective and scalable alternative to the expensive addition of base stations or access points to a wired infrastructure, as previously performed in many contexts (e.g. Home networking, Enterprise networking, etc.). Finally, the mesh backbone is resilient to node/link failures and unexpected events (e.g. link quality degradation, congestion), thanks to the capability of the backbone to provide redundant paths between any endpoint pair. Moreover, since a relevant part of the traffic is directed towards the Internet, most of the traffic sources have multiple destinations within the mesh network (i.e. multiple mesh gateways), which further increases path redundancy. The above benefits are some of the most representative of this novel networking paradigm, and give an idea on the reason behind the growing interest of the community on WMN deployments.

WMNs have been originally conceived as a low-cost extension of wired networks and intended for bulk data transfer. They have permitted to turn the ad hoc networking paradigm into a commodity, departing from the idea of isolated self-configured networks for specialized applications (e.g. military) in favor of a multipurpose networking platform that ensure Internet access at a low cost, meeting users' requirements to be connected *anywhere* and *anytime* (BCG05). The mixed architecture, combining fixed and mobile users with a static resilient infrastructure, has been regarded as a suitable solution to provide a cost-effective high-speed wireless connection in several diverse application scenarios. In the following, we briefly present some of the most common applications of

the wireless mesh networking paradigm. Interested readers may refer to (BCG05; AWW05) for a deeper discussion.

Broadband Home Networking: IEEE 802.11 WLANs are the common solution to provide broadband home networking. To guarantee the adequate service coverage, the installation of multiple access points is essential, although it is expensive and not practical since all the traffic between different access points has to pass through the backhaul network access hub. On the other hand, mesh routers can simply be located where service coverage needs to be extended, requiring a connection to the wired infrastructure only for a restricted subset of them. Wireless mesh networking avoids dead zones by positioning mesh routers as needed, and relying on the mesh backbone for internal communication, thus passing through the network access hub only when Internet access is required. Hence, WMNs offer a more robust and flexible connectivity service.

Community and Neighborhood Networking: The typical community networking paradigm is based on cables and DSL to gain Internet access, and to the last-mile modem-router wireless connection. However, also in this scenario traffic has to pass through Internet even in case of community-shared information, thus underutilizing network resources. Moreover, adequate service coverage may require a considerable up-front cost. Wireless mesh networking permits to extend service coverage by installing additional mesh routers and exploiting the robust multi-hop transmissions provided by the wireless backbone. Thus, even a large community can fully benefit from a limited number of Internet access points.

Enterprise Networking: A small network within an office can be considered as a home network, whereas a larger network connecting several offices or buildings represents a scenario similar to the community network described above. Hence, the basic principles provided earlier can be applied to motivate WMN deployment to provide flexible and resilient connectivity service for public and commercial networking, e.g. enterprises, airports, shopping malls. This

environments arise additional issues due to the size and complexity of the network topology. However, mesh routers deployment permits to increase service coverage and communication robustness keeping the cost limited.

Metropolitan Area Networks: Scalability is clearly one of the main issues to be considered in these scenarios. Usually, cellular networks are used to cover most of the large metropolitan area, offering lower rate than IEEE 802.11-based wireless networks and imposing high infrastructure-related costs. Cables or optical networks provide high-speed connections but require additional wireless devices to provide Internet access. Wireless mesh networking offers reasonable transmission data rates with a limited up-front investment, and can easily grow as the metropolitan area size increases. Moreover, it permits to provide connectivity even in environments where the set of Internet access points is very small.

Transportation Systems: In addition to Internet access points installed all over the city, municipalities and transportation companies are often interested in deploying an integrated system able to provide real-time information to passengers. Not only stations and bus stops need connectivity services, but also the vehicles can be provided with wireless technology. Clearly, low cost and extensive service coverage can be easily achieved with mesh routers deployed all over the transportation system, e.g. buses, trains, stations.

Security/Medical Systems: Both medical centers and surveillance systems need efficient monitoring, which usually requires to exchange a high and constant volume of data, due to images and videos. Also in these scenarios, WMNs offer an economically viable solution to provide the adequate service coverage without incurring the extremely high cost of the wired infrastructure.

Disaster Recovery: One of the most attracting feature of WMNs is the ability to provide resilient connectivity services and Internet access even in critical scenarios, where infrastructure is very limited and

deployment can not be planned in advance. Wireless mesh routers can quickly be deployed even in emergency situations, promptly offering a network that can significantly help in recovering from disasters, for instance by providing a real-time view of the scene. Although this is a typical scenario for MANETs, WMNs offer clear advantages over that networking paradigm.

Due to their attractive features as well as the wide range of possible application scenarios, WMNs have received increasing attention and stimulated a large body of research activities. Indeed, WMNs inherit most of the traditional challenges of ad hoc networks (CCL03). In particular, it is widely recognized that performance and reliability of wireless multi-hop communications significantly depend on the ability of the routing protocol to properly select network paths, given the current network conditions. A natural design approach for dealing with the complexities of the routing problem is to simply apply to the mesh domain the routing paradigms traditionally conceived for wired networks. This design choice implicitly assumes that wireless links are similar to wired links, and that they can be represented as point-to-point connections. For example, most of the routing schemes proposed for generic ad hoc networks (such as DSR (JHM07), AODV (PBRD03) and OLSR (CJ03)) select a shortest path between a source and destination pair, and forward each packet through a predetermined sequence of network devices, while assuming that link-layer retransmissions provide a reasonable level of communication reliability. Henceforth, we refer to this category of networking protocols as *legacy routing* solutions. However, wireless links are fundamentally different from wired links. First of all, the wireless channel is an intrinsic *broadcast medium* that has not clearly observable boundaries outside of which nodes are always unable to communicate. This implies that wireless links with intermediate packet loss rates, even higher that 50%, are quite common in typical outdoor mesh environments (ABB+04; CRSK06). Furthermore, wireless medium has time-varying and asymmetric propagation properties due to a variety of phenomena, including interference from external signals, wireless propagation impairments and fading (Rap02).

The above considerations on the peculiarities of the wireless communications suggest that, in order to improve the performance of WMNs, it is necessary to consider link qualities when choosing the best route between a source-destination pair. Indeed, a large body of research has been carried out in this area and different routing metrics have been proposed. The first metric proposed for wireless mesh networking is the ETX (DCABM03), which defines the cost of a link between a node and one of its neighbors as the expected number of transmissions that node requires to successfully deliver a packet to its neighbor. However, the implementation of this metric has shown poor performance in multi-rate environments, and an extension, called ETT (DPZ04), has been proposed, which defines the link cost as the time a data packet requires to be transmitted successfully. On the other hand, recent work has established that to correctly represent the quality of a link in a multi-hop environment, a routing metric should be able to capture other aspects of the wireless domain, such as the location-dependent nature of the link-layer contention (for instance, see CATT (GS08) and ETP (MBLD07) proposals), or the inter-flow and intra-flow interference (e.g., IRU (YWK06)).

Some of the proposed link-aware routing metrics have been implemented and tested in real network deployments, and experiments have shown that they can achieve significantly higher performance compared to a classical shortest-path routing algorithm. However, all these legacy routing protocols *pre-compute* one or more minimum-cost paths (see, for instance, multi-path schemes described in (MGLA05; GK04)) for each source-destination pair. Experimental evidence (BM05; KRH$^+$08; CJKK07) has also proved that using predetermined paths can be ineffective in dealing with unreliable and varying wireless environments. For these reasons, recently researchers have been investigating radically new routing approaches, which exploit the multiple transmission opportunities that the broadcast nature of the wireless medium creates. More precisely, whenever a packet is transmitted, it is simultaneously received by multiple nodes, which may experience significantly different channel conditions. This property is called *multi-user diversity* because it refers to a type of spatial diversity existing across multiple receivers (or

users) (DADSC04; QB03). This intrinsic diversity of the wireless environment is not a drawback per se, but it may cater for new design principles and alternative routing paradigms. Several protocols can be included in this novel class of routing strategies that exploit receptions of the same packet at multiple nodes to increase network performance compared to legacy routing. In this chapter, we give a comprehensive review of two of the most promising design approaches: *opportunistic forwarding* and *network coding*.

Opportunistic routing algorithms implement forwarding decisions in a hop-by-hop fashion, and they defer the selection of the next hop for a packet until they have learnt the set of nodes which have actually received that packet (BM05). This permits to optimize the selection of the packet forwarder(s) and to discover on the fly the best network path. This strategy clearly departs from the design principles of legacy routing, which assigns a predetermined next hop to each packet. It is also important to note that the term "opportunistic" refers to a wider class of routing algorithms based on the common idea of leveraging any transmission opportunity rather than imposing the packet transmission along a predetermined path. For instance, opportunistic routing is also used in intermittently connected networks (PPC04). However, in that context, communication opportunities are generated by mobility, which enables pair-wise contacts between nodes. In contrast, in this chapter we limit ourselves to static networks, where transmission opportunities rely on the variability of channel conditions and on the broadcast nature of the wireless medium.

The second design principle we analyze in this chapter is wireless network coding, which allows the network nodes to combine/encode the data packets they receive, so as to compress data information and to increase the innovative content carried into each packet (KRH$^+$08). At the same time, network coding may increase reliability of packet transmissions because each encoded packet mix information about multiple packets, thus increasing the probability that they would reach their destination. It is also useful to note that the boundary between network coding and opportunistic forwarding may be blurred in some cases, when

both approaches are jointly used. In these cases, we will prefer the term *hybrid* routing to point out that network coding and opportunistic forwarding are integrated into a unified routing scheme (CJKK07).

The above discussion provides only a brief insight into the reasons of performance gains achievable with opportunistic forwarding and network coding. The objective of this chapter is to analyze in a thorough way the various conditions in which these two routing paradigms may provide the most significant performance improvements.

2.2 Background and Taxonomy

In this section, we overview the general routing approaches that can be adopted to take advantage of opportunistic forwarding and network coding in WMNs. Specifically, we introduce three main routing categories and several related sub-categories. Then, we describe the representative features, benefits and design challenges of these three classes of routing approaches.

2.2.1 Opportunistic routing

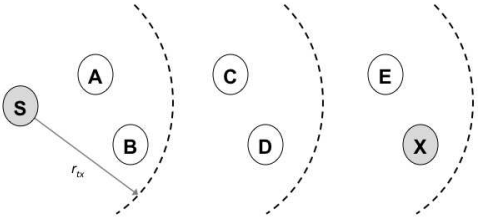
The opportunistic-based routing concept considered in this study is characterized by two main features: $i)$ any node overhearing a packet transmission is involved in the forwarding process, and $ii)$ the selection of the next forwarding node(s) is deferred after packet reception (BM05). As previously explained, legacy routing algorithms rely on transmitters that select one or more designated next hops before delivering the packets, which implies that each packet must know a priori its next relay(s). However, this design principle borrowed from the routing protocols for wireline networks, does not appear suitable for wireless networks. Indeed, it masks the broadcast property of wireless communications under an artificial point-to-point link abstraction (DADSC04). On the contrary, opportunistic routing fully embraces the broadcast nature of wireless medium because whenever a node is willing to deliver a packet, it performs a broadcast transmission and, then, the nodes that successfully

receive the packet autonomously select the "best" next forwarder. This allows each packet to dynamically construct the optimal route to reach its intended destination according to the link conditions at the time a packet transmission is performed.

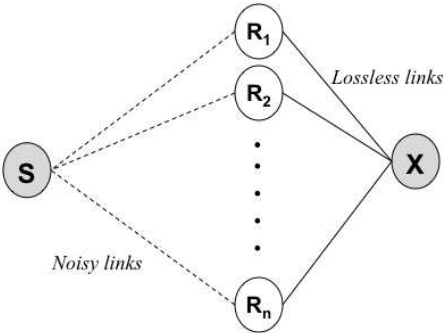
Intuitively, the main benefit of opportunistic-based routing is to leverage transmission opportunities that unexpectedly reach far nodes, taking advantage of any transmission progress while mitigating the negative impact of failed transmission attempts. Moreover, this strategy allows the destination node to receive packets that have been forwarded by different relays, thus traversing different network paths, fully exploiting the multi-user diversity property. In contrast, legacy routing would require retransmitting any packet that does not reach the next hop for which it was intended, as well as the preliminary construction of one or multiple network paths connecting the source/destination pair.

For a better understanding of the inherent benefits associated to opportunistic forwarding, let us consider the simple topology illustrated in Figure 2(a), firstly analyzed in (BM05). Let us suppose that the best route from source node S to destination node X selected by a traditional shortest-path routing is S-B-D-X. If a packet sent by S is correctly received by node A but not node B, then it has to be retransmitted by S until it reaches the intended next hop B. This is the case of a transmission falling unexpectedly short. Another possible situation is a packet sent by S that is correctly received by both node B and node C. Although node C is closer to packet destination than node B, it is not allowed to relay the packet. On the contrary, opportunistic routing techniques take advantage of any of these situations to maximize the progress towards the packet destination that each transmission may provide. Moreover, retransmissions are avoided whenever possible, i.e., if there is any alternative forwarding possibility. Thus, by avoiding wasting of network resources through useless transmissions, it is possible to significantly increase the overall network throughput.

Another interesting benefit of opportunistic routing is the ability of combining many weak physical links into one stronger *virtual* link. As shown in Figure 2(b) for a diamond-shaped topology, the sender has a



(a) Linear topology



(b) Diamond topology

Figure 2: Illustrative topologies clarifying the inherent benefits of opportunistic-based forwarding.

low delivery probability to all its n neighbors, while they have a high probability to successfully deliver packets to the destination. In this setting, it may be more advantageous to broadcast a packet in order to increase the probability that at least one of the possible one-hop neighbors correctly receives it. After the packet reception, the "best" node among those that received the packet will be responsible for further forwarding it to the destination. On the other hand, legacy unicast routing strategies would select in advance one of the available neighbors as the exclusive relay for the communications between the source and the destination. This may lead to many retransmissions before the source would be able to successfully deliver a packet to the intended next hop.

At this point, it should be clear that this capability of taking advantage of any transmission opportunity that arises in the network is the principal root of the performance improvements provided with opportunistic forwarding. However, this flexibility comes at the cost of an increased design complexity. In particular, the key technical challenges to be addressed when designing a new opportunistic protocol for WMNs are the following.

How to select the next forwarding node(s)? In principle, all the network nodes may cooperate in the forwarding process, and be considered as candidate relays. The selection of the next forwarding node(s) among the candidate relays should maximize the transmission benefits, measured in terms of the selected performance metric (e.g., reliability, throughput or the end-to-end delay of the flow). However, the selection of the best forwarding node(s) requires the implementation of a *coordination* process among the candidate relays, which may require explicit exchange of state information. It is intuitive to note that the coordination overheads and complexities increase with the number of candidate relays involved in the coordination process. For these reasons, most of the existing solutions for opportunist-based routing relax the constraint of "pure" opportunism, and assume that the flow source specifies in advance a *subset* of *candidate relays* for each packet or block of packets, which are the only nodes allowed to participate in the forwarding process. Various schemes have been proposed for the selection of candidate relays

based on some notion of "*closeness*" of the nodes to the packet destination (BM05; DFGV07; ZWNL06). In general, these mechanisms assume that there is an underlying link-state routing protocol that constructs a map of the link qualities in the network, permitting to compute the approximate cost of using a node as forwarder to reach the intended packet destination (according to a given routing metric). Then, the identities of the selected candidate forwarders are listed in the packet header. Thus, whenever a node receives a packet, it first checks if it is in the forwarder list of that packet, in which case it further processes the packet; otherwise it discards it. This strategy keeps the coordination overhead limited, as far as the number of participants is kept small. To further facilitate the selection of the next forwarding node(s), the list of candidate forwarders may be also ordered by assigning to each node a fixed *priority* value to be used during the forwarding process. A common design choice is to derive the node priority from the distance between that node and the packet destination. In the following, we will further elaborate on the role of prioritization in the implementation of an opportunistic forwarding process

When relays should forward a packet? After selecting the candidate forwarders, it is necessary to establish the time at which the packet should be forwarded. In fact, differently from classical routing, where packet forwarding at the next hop node should immediately follow the reception of the packet, in general opportunistic routing solutions introduce forwarding delays. The main reason for this design choice is that imperfect coordination among candidate relays, as well as packet losses, may cause multiple duplicate transmissions of the same packet from multiple nodes. To avoid this unnecessary waste of network resources, a simple solution is to establish a *scheduling* among the candidate relays, and to set different forwarding timers at the selected forwarders. For instance, many schemes use the differentiation/prioritization of the candidate relays to assign fixed and constant forwarding timers to each of the potential forwarders (BM05; RSMQ06). Although the overhead for such a scheduling is high, each node may be aware of its own forwarding time, without the need of a real-time agreement with the other nodes. Then,

overhearing of other nodes transmissions or explicit exchange of state information can be used to cancel transmissions of packets already delivered by higher priority nodes. To summarize, scheduling techniques ensure the effective suppression of most duplicate transmissions at the cost of an increase in packet delays and protocol complexity, which are necessary to establish node scheduling without requiring explicit signalling before each transmission. An alternative approach more appropriate for delay-constrained traffic is to implement randomized schedulers, permitting each node to probabilistically decide if continuing forwarding the packet towards the destination (YYW+05). However, in this case, additional mechanisms are needed to control the level of redundancy in packet transmissions, such as rate control mechanisms or methods to limit the maximum number of forwarders for each packet traversing the network.

How to acknowledge packet reception? Broadcast transmissions are the basis of any opportunistic routing scheme. However, broadcast frames do not implement link-layer acknowledgements. Thus, acknowledgment mechanisms should be introduced, either at the routing or link layer, to provide transmission reliability. Two options can be considered: end-to-end acknowledgements generated by the final destination (BM05), or hop-by-hop acknowledgements generated by the forwarders (RSMQ06). The former reduces overhead but it may lead to higher delay since the forwarding progress depends on acknowledgements generated by the destination. The latter is in contrast with the broadcast transmission principle, even though it contributes to reducing delay and ensuring correct packet reception. Note that for schemes operating on blocks of packets rather than individual packets, acknowledgment information can be easily grouped together, limiting the number of needed acknowledgment messages. Usually, in this case map-based approaches are adopted to implement selective acknowledgment for group of packets (BM05). However, these techniques add significant overhead to packet headers, and they require a careful design and tuning.

How to control congestion? Generally, most of the link-layer technolo-

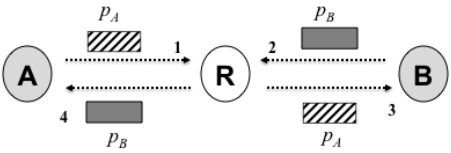
gies transmit broadcast frames without using congestion-aware access methods. Furthermore, duplicate transmissions and multiple flows may exacerbate the collision problem. Thus, the surge of congestion is a critical aspect to be taken into account during the forwarding process. This is still an open issue because most of the current work on opportunistic routing has focused on the design of mechanisms to avoid redundant transmissions, rather than controlling the broadcasting rate. For instance, credit-based schemes have been proposed to control the spreading of packets throughout the network (YYW+05), or basic window-based techniques have been used to limit the rate with which new packets are injected in the network (RSMQ06). However, opportunistic routing solutions generally forward data packets based only on link conditions and routing metrics that reflect the cost to reach the packet destination. On the other hand, the forwarding process should take into account typical burstiness of data flows and the contention among multiple sessions (RSMQ06). This would require up-do-date congestion information to be spread across the network, which arises additional issues.

Keeping in mind the main points discussed above, we believe that a key aspect in the design of opportunistic routing protocols is the strategy adopted for the coordination of the candidate relays during the forwarding process. In our view, this is an appropriate aspect to be considered when classifying existing solutions. Thus, we divide opportunistic routing schemes into *scheduled* and *not-scheduled* algorithms. The first category of solutions identifies a prioritized subset of potential forwarders and specify their scheduling. This list specifies not only the nodes allowed to participate in the forwarding process, but also the order in which they have to transmit, thus their scheduling. On the contrary, not-scheduled schemes allow each node to autonomously decide whether to forward a packet and when to do it. Clearly, a subset of potential forwarders may still be provided in order to simplify the forwarding process, but without establishing in advance a prioritization. The most representative schemes belonging to the above two sub-categories are briefly described in Section 2.3.1.

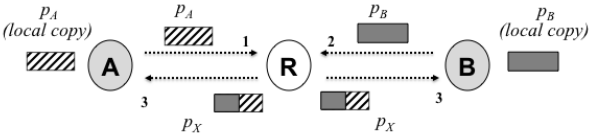
2.2.2 Coding-based routing

The basic principle behind network coding is that routers can combine the information to be transmitted so as to deliver multiple data packets through a single transmission. More precisely, let us denote as *native* packets the original non-coded packets that are initially generated by the source node. Then, a *coded* packet is a combination of the native packets, which the destination node can decode to reconstruct the set of initial packets. Potential advantages of network coding were first demonstrated in the pioneering paper by Ahlswede et al. (ACLY00), which considers multicast transmissions in wired networks. In that paper it is shown that network utilization can be enhanced if network nodes do not act as classical switches, i.e., routing or replicating packets, but as encoders that mix the information they receive from all the input links and send it to all the output links. The authors demonstrate that network coding allows to achieve the multicast capacity, which is the maximum rate at which a sender can communicate common information to a set of receivers. Moreover, Li et al. (LY03) show that linear coding is sufficient for the above condition to hold, while Ho et al. (HMS$^+$03) demonstrate that this is true also when nodes pick random codes.

Network coding benefits are not confined to multicast transmissions in networks with point-to-point links. Network coding techniques naturally extend to wireless networks by taking advantage of the broadcast nature of the wireless channel (FKM$^+$07). In wireless networks, nodes can overhear neighbors' transmissions. Hence, each node may be able to collect many packets to be coded together, thus increasing the efficiency of the forwarding process in many cases. To better explain the performance gains obtained by employing network coding techniques in the context of wireless networks, in the following we illustrate a simple coding example. To this end, let us consider the chain topology depicted in Figure 3(a), where node A wants to send packet p_A to node B, and node B wants to send packet p_B to node A. In this case, intermediate node R must forward both packets received by node A and node B because they can not directly communicate to each other. Thus, with legacy routing,



(a) no coding



(b) coding

Figure 3: A simple example of achievable coding gain in wireless environments.

four transmissions are needed in order to deliver one packet to both destinations. On the other hand, as shown in Figure 3(b), network coding allows node R to broadcast a single coded packet, say p_X, generated by applying the *XOR* operator to the native packets (i.e., $p_X = p_A \oplus p_B$). Then, node A can easily recover packet p_B since it locally stores a copy of packet p_A and $p_A \oplus p_X = p_B$, while node B can reconstruct packet p_A with analogous operations. In this way, three transmissions are required instead of four, with a 33% improvement of network capacity. Note that the overall coding gain depends on both the network topology and the traffic patterns. For instance, a similar reasoning can be applied to a cross-based topology, in which four flows intersect the central node R. Then, node R can combine the four packets received by its neighbors into one coded packet. Assuming overhearing among the neighboring nodes, four nodes are able to exchange packets in five total transmissions instead of eight, with a 60% improvement of network capacity. Hence, coding gain is more significant in larger networks, where more *coding opportunities* arise. In general, a coding opportunity may be defined as the

possibility of creating a coded packet that can be successfully decoded from the intended destinations of the native packets.

Another important feature of network coding is the ability of providing reliability with low complexity, which is particularly relevant in lossy environments. Indeed, packet loss is a not negligible issue in wireless networks, and retransmissions are the simplest method to ensure reliability. However, simplicity comes at the cost of increased congestion and higher collision probability due to retransmitted packets. In this context, network coding offers a more convenient alternative to retransmissions of original packets by spreading information about several packets through their combination. This leads to a certain level of reliability in a more efficient manner. Even in case of packet loss, nodes may still be able to recover the original packets without asking for any retransmission.

Although the basic principles of network coding are intuitive, there are critical aspects to be considered when developing a coding-based routing solution, which can be summarized as follows.

Which packets should be coded together? To maximize coding gains it is necessary that each node receiving coded packets is able to recover all the original native packets. This goal can be achieved by imposing some constraints on coding decisions taken by every node. The coding process regards *which* packets must be coded together and *how many* coded packets must be sent (i.e., how much *redundancy* must be guaranteed). A primary basic distinction is between intra-flow and inter-flow coding styles. If a network coding technique is intra-flow, then each node must encode packets together only if they belong to the same flow (GHK$^+$07). Thus, packet selection is mainly driven by the flow membership. On the other hand, if a node can select packets intended for different next hops, the choice is more complex. Whenever a node is willing to send data, it must select the subset of native packets that maximize a certain metric, which should reflect the possibility for each neighbor to recover native packets (KRH$^+$08; RSW$^+$08). As explained later, this is strictly related to the encoding scheme used to code packets together. Finally, several techniques have been proposed to improve encoding efficiency. For instance, a common approach is to group packets into blocks and to permit only

the coding of packets belonging to the same block. This solution aims to find a trade-off between network coding benefits and complexity.

How to code packets together? Computational complexity is a crucial issue in network coding, and selection of coding techniques must consider the impact both in terms of encoding and decoding complexity, and in terms of minimum number of coded packets needed to recover the original flow. In the example presented above, coding is performed through *XOR* operations, which are easy to implement. However, the most frequently used approach for encoding packets is through *random linear codes* (LY03; HMS+03). Specifically, given a set of k native packets $p_1, \ldots, p_k$, the coded packet p' can be created as $p' = \sum_{j=1}^{k} c_j p_j$, where c_j are random coefficients extracted from a certain finite field $\mathbb{F}_q$[1]. Random linear codes have some nice properties. First of all, checking for independence between coded packets requires only simple matrix algebra, and decoding can be done inverting the matrix of coding vectors. Furthermore, a linear combination of coded packets is also a linear combination of the corresponding native packets, which greatly simplifies the re-encoding process at intermediate forwarders. Several theoretical studies on properties of random linear coding have been conducted, demonstrating the potentiality of this technique, both in lossless and in lossy environments (LMK05).

When coded packets should be generated? Many factors affect the selection of the time at which coded packets should be generated. In general, a coded packet should be created only when there is a coding opportunity, i.e., the node has enough packets to code together. However, a node may have packets to send but no coding opportunities, thus it may decide either to forward native packets or to further delay transmissions waiting for receiving additional packets. Clearly, this design choice represents a trade-off between delay and achievable coding gain. Note that also buffer constraints must be taken into account to decide how long packets useful for encoding should be stored by each node. Fur-

[1] A finite field $\mathbb{F}_q$, or Galois Field $GF(q)$, contains a finite number q of elements, where $q = p^n$, p is a prime number and n is a positive integer. In general, for network coding purposes $p = 2$.

thermore, the coding algorithm should ensure that intermediate nodes have received enough coded packets to decode their corresponding native packets (KRH+08; NSZN06). For instance, intermediate nodes may want to reconstruct native packets to refresh the packet stream by replacing coding coefficients and re-encoding incoming packets. In addition, coded packets may include packets from multiple flows, and intermediate nodes may want to decode incoming packets to avoid that data is forwarded to areas where there are no interested receivers. Thus, the broadcast rate of coded packets should be adjusted to ensure that the decoding probability is sufficiently high not only at intended destinations but also at intermediate forwarders. Finally, transmissions of coded packets can also be driven by a trade-off between the desired level of data redundancy and the achievable coding gain (RSW+08). Specifically, coding is generally used to minimize the total number of transmissions needed to carry packets across each wireless hop. However, in case of high loss rates, it may be desirable to increase redundancy by injecting more coded packets in the network, so as to ensure that the next hop forwarders receive enough packets to be used during the decoding process, even at the cost of increasing the number of transmissions required to communicate the same information.

Several solutions exist to deal with the various issues described above, and to fully exploit the coding benefits. We believe that the key characteristic pertinent to network coding that can be used to discriminate between coding-based routing schemes is the set of rules employed to decide which packets code together. To this end, the distinction between the two complementary approaches of *intra-flow* and *inter-flow* network coding, represents an essential principle for the network coding classification. The most representative schemes belonging to these two sub-categories are briefly described in Section 2.3.2.

2.2.3 Hybrid routing

From the above discussion we can conclude that opportunistic forwarding and network coding are two complementary means of taking ad-

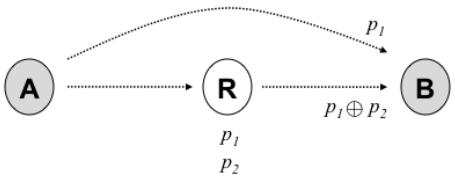


Figure 4: An example of potential advantages of hybrid schemes.

vantage of the broadcast nature of wireless channel, as well as to exploit the multi-user diversity of typical wireless environments. It is also intuitive to anticipate that coupling these two approaches into a *hybrid* paradigm may permit to obtain significant improvements, originated by joining advantages of both techniques. Moreover, this coupling can offer an implicit solution to some limitations of the two paradigms. For instance, one of the main issues of opportunistic routing is the scheduling overhead for node coordination. Classical opportunistic forwarding deal with this issue by introducing node prioritization and forwarding delays, or exchanging state information between candidate relays. In contrast, network coding may provide an elegant method to partially eliminate this complexity. In principle, nodes do not need to know exactly which packets are stored by each neighbor and which packets are sent by the other forwarders. Indeed, if n native packets have to be sent, then any set of n different coded packets is sufficient to recover the original set. Hence, every forwarder may autonomously generate its own coded packets, since any of them contains information about several native packets, and it may contribute to the flow progress towards the destination. Clearly, this solution carries also some network coding issues to the opportunistic setting. In theory, each forwarder can create and broadcast coded packets, but this may lead to a high number of unnecessary transmissions. A possible solution is to allow node to code and forward only *innovative* packets. However, the formulation of the innovative property depends on the specific scheme, and we describe it in details in Section 2.3.3.

In order to better clarify the advantage of the hybrid paradigm, let us explain it with an example. In the chain topology shown in Figure 4 source node A should send two packets, p_1 and p_2, to the destination node B. Let us assume that a routing protocol has selected the two-hop path A-R-B because the direct communication A-B is too "weak", e.g., affected by a high loss probability or using a slow transmission rate. Thus, packets p_1 and p_2 will be sent to intermediate node R, which should further relay them to the intended destination. Now, let us assume that relay R receives both packets p_1 and p_2 correctly, and that node B can directly overhear at least packet p_1 from node A. With legacy routing p_1 correct reception at node B is useless, because the packet is discarded. In contrast, opportunistic routing permits to take advantage of this unexpected reception, reducing the number of packets that relay R must forward to node B. However, if only the opportunistic paradigm is employed, node R would first need to communicate with B to know which packet(s) it misses and, then, send it(them). On the other hand, by exploiting coding techniques, R can broadcast linear combinations of the two packets, allowing B to recover the missing packets potentially requiring a smaller number of transmissions and without exchanging additional control messages. For instance, if R broadcasts $p_1 \oplus p_2$, then only one transmission from R is sufficient to successfully complete data exchange. Clearly, after recovering the missing packet, B has to send an ACK to notify R of its successful reception, as it is required also in the legacy case. In summary, this basic case illustrates the main principle of a hybrid scheme. Nodes perform *broadcast* transmissions of *coded* packets without having a designated next hop. Clearly, specific details depend on the chosen approach, but the above considerations hold in general.

When designing a hybrid scheme, most of the technical issues are inherited from the individual techniques. Below we focalize only on the critical issues that are specifically related to the hybrid paradigm.

Which packets should be coded together? Although this issue has been already discussed in the network coding context, it should be revisited by taking into account the peculiarities of opportunistic forwarding. More precisely, using legacy routing each packet transmission has a designated

next hop node. In this case, the coding process may easily discover, through probabilistic considerations and overhearing or explicit signaling, which are the packets available at the different destination nodes. Thus, each relay node can locally decide which coded and native packets should be transmitted to maximize some metric (e.g., throughput, packet delivery rate, etc.) across all the intended next hop nodes. In contrast, when the path is constructed hop-by-hop at the packet recipients, the concept of designated next hop is not valid anymore, thus different coding strategies should be employed. In general, when opportunistic routing and network coding are used together, the aim should be to spread coded packets across the network in order to increase coding opportunities rather than sending (native) packets to a specific subset of nodes (CJKK07). To simplify this coding decision process, intra-flow coding is commonly used in existing hybrid routing schemes.

When a node should stop sending packets? From the point of view of opportunistic routing, a relay node should keep transmitting a packet until it is sure that at least one node closer to the destination has received it. On the other hand, network coding imposes a more demanding constraint because a minimum number of independent coded packets must be received at the destination node for ensuring correct decoding (CJKK07). In principle, a forwarder may keep transmitting stored packets to increase levels of redundancy and improve successful decoding probability at the destination node. However, an efficient stopping rule is needed to achieve those goals while ensuring tolerable delays and overheads limited. To facilitate the protocol design, usually the coding process operates on *blocks* of packets[2]. Thus, the stopping rule reduces to the policy used to stop processing a certain group of packets and start with the next block. Two general approaches can be identified. The first idea is that the destination directly sends a message to the sender when it receives enough packets from a certain block, so that the sender starts processing a new block and informing all the other nodes to stop coding packets of the previous block (CJKK07). Thus, the destination drives

[2]The block here is intended as a group of consecutive packets belonging to a certain flow. In practice, each coding approach has its own "grouping" policy.

coding decisions at each hop, according to its decoding goals. An alternative strategy entails that every node is in charge of ensuring decoding at its own neighborhood. The key point is that the source and all the intermediate nodes move from one block to the next one based on the current situation of their neighbors, thus each node contributes to ensure the local complete and correct reception of a block, which in turn guarantees correct decoding at the destination itself (KHW08). The details of different solutions and possible enhancements are discussed in Section 2.3.3.

The classification of hybrid based routing solutions is slightly trickier than in the other two routing categories previously presented. Clearly, key features are derived from the basic "components", since a hybrid solution is intrinsically opportunistic and coding-based. Thus, in principle we could think to derive a taxonomy for this class of solutions from the same criteria used for its building blocks. From the opportunistic side, the categorization in *scheduled* and *not scheduled* approaches loses its validity in the hybrid case, since network coding partially eliminates the need of node scheduling. Thus, as for coding-based approaches, we could opt for a classification based on flow-membership constraints in the native packets selection process. However, as stated above, inter-flow coding is perhaps harder in a hybrid context, where the notion of predetermined next hop node is not valid. In our vision, hybrid based routing can be better classified based on the used stopping rule. Specifically, we categorize as *destination-based* the hybrid schemes that rely on the destination to drive coding decisions at each hop. In contrast, we denote as *neighborhood-based* the hybrid schemes that move the focus of coding decision to the neighborhood rather than the destination. Hence, the key difference between *destination-based* and *neighborhood-based* coding is that the former requires notifications from the destination about its current situation, while the latter rely on local context discovered through overhearing and, in some cases, explicit signalling. The most representative schemes belonging to these two sub-categories are briefly described in Section 2.3.3.

2.3 Overview of representative solutions

In this section, for each of the three routing categories analyzed in Section 2.2, we overview the design choices and operations of some of the most representative schemes. For the sake of clarity, Figure 5 illustrates the proposed taxonomy and lists the solutions that will be analyzed more in details in the following.

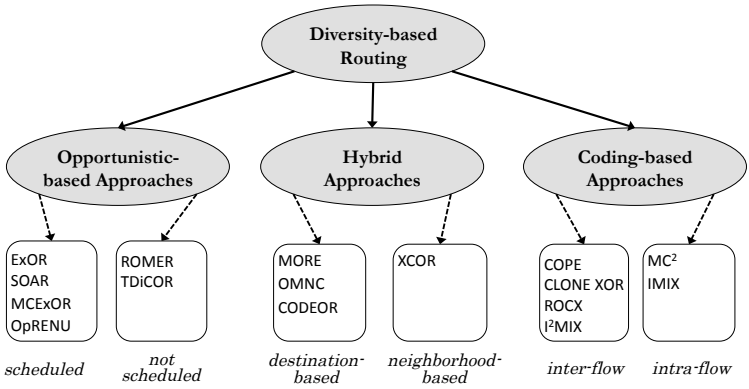


Figure 5: Taxonomy of routing approaches for WMNs taking advantage of multi-user diversity.

2.3.1 Opportunistic routing

As reported in Figure 5, opportunistic-based routing solutions are divided into two sub-categories: *scheduled* approaches and *not scheduled* approaches. The following two sections outline the most relevant schemes proposed for both approaches.

Scheduled schemes

Schedule-based opportunistic routing originates from the seminal paper from Biswas and Morris, who have proposed the *Extremely Opportunistic Routing* protocol (*ExOR*) (BM05). To reduce the coordination overhead between candidate relays, in ExOR the packets to be transmitted are grouped into batches according to their destination node. For each packet of the same batch, the source node selects a subset of optimal candidate forwarders, which are prioritized by closeness to the destination. The closeness property of a node is evaluated employing the ETX metric (DCABM03), i.e., estimating the average number of transmissions needed to reach the destination from that node along the lowest-ETX path. Thus, the implicit assumption underlying ExOR design is that a link state routing protocol is also running in parallel to the opportunistic routing to efficiently collect links' delivery probabilities.

The list of selected forwarders, ordered by node priority, is added to the header of each packet broadcasted by the source. Hence, each node receiving a packet knows whether it has to participate in the forwarding process or not, and its position in the forwarding schedule. Due to inter-node loss rates, each candidate forwarder will successfully decode only portions, called *fragments*, of the packet batch it has received. In order to distribute information on which fragments each forwarder has received and rebroadcasted, each packet also contain a *batch map*. For each packet in the batch, this map indicates the highest-priority node known to have received a copy of that packet (BM05). Then, as the packet progresses towards the destination, the batch map contained in the packet is used to update the local batch maps stored in the receiving nodes, which list the IDs of the node closest to the destination known to have transmitted that packet. A forwarder is allowed to broadcast only received packets that its local batch map indicates have not been forwarded by any other higher priority node. Moreover, to avoid simultaneous or duplicated transmissions by different nodes, whenever a forwarder receives a packet it sets a timer, called *forwarding timer*. This timer is an estimate of the time that would be necessary to higher priority nodes to transmit the

remaining packets in the batch. Then, only when the node's forwarding timer elapses, it can rebroadcast the packets it has received, and which its local batch map does not indicate as received by higher priority nodes.

It is now evident that the batch maps contained in the headers of transmitted packets play a crucial role in ExOR. On the one hand, they are used as a sort of gossip mechanism to disseminate reception information from higher priority nodes to lower priority nodes. On the other hand, batch maps are used also for the acknowledgement process. More precisely, when the destination receives a new packet it sends back to the source its batch map using a legacy unicast routing. In this way, the source knows when the destination has received most of the current batch, so that it can pass to transmit a new batch. However, ExOR only guarantees to transmit 90% of a batch using opportunistic forwarding, while the remaining packets are sent with legacy unicast routing.

The strict schedule ExOR establishes between candidate forwarders is somehow equivalent to a slotted polling system. As shown in (BM05) through experiments in an urban mesh trial, the ExOR scheduling is effective in ensuring that each packet is retransmitted a minimal number of times, and in limiting the probability that multiple forwarders rebroadcast the same packet. However, the simplicity of this scheme comes at the cost of its inefficiency. First of all, the largest the set of candidate forwarders and the longer is the cycle of the scheduler. Furthermore, since candidate forwarders can be out of each other radio range, ExOR fixes a minimum value for the forwarding timers. Thus, even if a node has no batch fragments to transmit, the scheduler blocks the lower priority nodes. Finally, the scheduling duration is not dependent on the number of packets to transmit. Thus, the scheduling overhead would be excessive for a relatively small number of packets. This is the reason why in the original ExOR design the last 10% of packets in a batch are routed to the destination using legacy link-state routing. In addition, ExOR mandates the use of large batches of the order of tens of packets. This also implies that ExOR works well only with persistent flows, which always generate the minimum number of packets needed to fill a batch. In addition, the use of ExOR-style scheduling with multiple

concurrent flows, which can induce conflicting forwarding timers, is not specified in (BM05). Nevertheless, ExOR approach has raised very high interest in the research community, where many groups started to work on solutions based on the same principle: to abandon the concept of predetermined paths in favor of paths constructed hop-by-hop.

An interesting variant of the ExOR solution is the *Simple Opportunistic Adaptive Routing* protocol (*SOAR*) (RSMQ06). Similarly to ExOR, SOAR employs a scheduling scheme relying on priority-based forwarding timers to avoid duplicate and simultaneous transmissions by different nodes, being ETX the metric used to estimate node's closeness to the destination, and its related priority. However, the strategy used by SOAR to establish the schedule among the candidate forwarders is radically different from ExOR. First of all, SOAR does not use batch maps to explicit signal among candidate forwarders on packets' reception status, but it employs overhearing to coordinate forwarders' transmissions. More precisely, whenever a node overhears a transmission from higher priority nodes, it will cancel its forwarding timer and remove that packet from its queue, thus avoiding duplicate transmissions. To ensure that the candidate forwarders are close enough to overhear each other with a high probability, SOAR avoids diverging paths and uses only network paths in close proximity to the shortest route between the source and destination. Moreover, SOAR abandons the use of packet batches and operates on individual packets. Since packets are not organized in batches, also the computation of forwarding timers is simpler in SOAR than in ExOR because the former can use constant timers proportional to the node priority, while the latter employed variable timers whose duration depends on the number of packets buffered in higher priority nodes and the receiving data rate.

Another aspect that differentiates SOAR from ExOR is the use of hop-by-hop retransmissions, which are driven by network-layer ACKs generated by the highest priority forwarder that received the packet. However, to increase the reliability of the forwarding process and minimize useless retransmissions, SOAR uses a combination of various ACK mechanisms, including selective ACK to acknowledge all recently received packets,

as well as piggyback ACKs and ACK compression to reduce ACK overhead. Finally, although SOAR does not employ packet batches, it allows each forwarder to transmit a new packet even if there are other outstanding unacknowledged packets. To this end, SOAR uses a classical sliding-window protocol to control the maximum number of outstanding data packets. Note that in (RSMQ06) it is proposed to use a small window (only three packets) to limit the transmission delays.

The *Multi-Channel ExOR* protocol (*MCExOR)* (ZKR07) apply opportunistic routing to multi-channel wireless networks. Similarly to ExOR, this protocol uses a prioritized set of candidate forwarders. However, the multi-channel extension requires the computation of one set for each radio channel. To decouple routing from channel assignment, in MCExOR it is assumed that the assignment of channels to nodes is carried out independently of packets flows. In this case, each node is simply characterized by its *home channel*, i.e., the channel it is operating on.

Since MCExOR leverages the ExOR design principles, the most important task it has to perform is the selection of the candidate forwarder set. However, while ExOR employs a simple and centralized selection rule, i.e., a candidate forwarder is *any* node in the network that would be able to transmit at least 10% of the packets in a batch, MCExOR defines a more sophisticated and localized heuristic. First of all, in MCExOR a source of a data flow selects among its neighbors the nodes that have an expected cost of delivering a packet along the lowest-ETX path to the destination lower than its own. Then, the selected neighbors are grouped according to their home channels. Finally, among each group, all the possible combinations of candidate forwarders are considered. Each of these subsets has an associated cost, which depends on the average number of transmissions that would be needed to reach the destination in case that set of forwarders would be used[3].However, the optimal set is not the one that simply has the minimum metric, but the one that also minimizes self-interference, which is caused by the use of the same radio channel at each hop along the path. Thus, a multi-channel environment introduces

[3]To compute the optimal set in a more efficient way, MCExOR assumes that only the first hop in path is opportunistic.

a further dimension in the routing process because the goal now is to reduce not only the number of data transmissions, but also interference among packets belonging to the same flow.

Once the source has selected its set of best forwarders, it broadcasts the packet, whose header contains the forwarder list. To select which candidate forwarder must carry on with packet transmission, MCExOR relies on slotted link-layer acknowledgments. Specifically, candidate forwarders that received the packet send their ACK in order of decreasing priority, all separated by a delay of *SIFS*. In case of ACK missing, other nodes willing to send data may sense the medium idle for a *DIFS* period, which allows it to start a new transmission. Thus, in order to avoid collisions, the mechanism is defined as a compressed slotted acknowledgement, where each node sends prematurely its own ACK whenever it detects a missing acknowledgment from the previous (higher priority) forwarder. Obviously, the highest-priority candidate forwarder to have sent the ACK is elected as the next relay node. Then, the new forwarding node must identify its optimal set of candidate forwarders applying the same algorithm used by the source. Thus, differently from ExOR, in MCExOR the candidate forwarder list is recomputed after each transmission by the node that has been selected to rebroadcast the received packet. It is intuitive to note that this is possible only if MCExOR operates on individual packets, as SOAR, and not on packet batches.

A different approach from the ones presented so far is introduced in (WLL08). The basic idea is that nodes closer to the destination are not the only valid potential forwarders, as in many other opportunistic solutions, and an utility framework is proposed to estimate the benefit of the successful delivery of a packet. More precisely, a benefit value is attributed to each packet originated at source S and heading to destination D. Then, an expected *utility* value can be associated to the packet delivery on each multi-hop network path between the source and destination, computed as the packet benefit minus the path cost. It is also possible to compute the *residual expected network utility* (*RENU*) for each node on the network path, which represents the utility to use that node as relay for packet destination D. In other words, RENU parameter re-

flects the node's closeness to the destination in terms of utility. Then, the optimal route that maximizes the utility will depend not only on the topology, but also on the chosen benefit value, which is a unique property of utility-based routing (LW06). In case of opportunistic routing the utility metric is reformulated as *OpRENU* (WLL08) to take into account that there are multiple candidate forwarders and not a single next hop. In other words, the utility of a node depends not only on the utility of that node selected as relay to reach the destination, but also on the utility of all the nodes belonging to the same relay set. Due to this mutual dependency, determining the optimal relay set that maximizes the network utility is a complex problem that requires an exhaustive analysis of all paths from source to destination. Thus, in (WLL08) an heuristic solution is proposed to select relays and to determine priorities among them. With this heuristic, the relay selection procedure operates only on a restricted subset of the possible paths, according to the order obtained through RENU values.

Not scheduled schemes

In contrast with the approaches described above, the family of not-scheduled schemes mitigate the design complexity of opportunistic routing by avoiding strict scheduling among candidate relays. One of the first examples of such approach is the *ROMER* protocol (YYW+05), which introduces a credit-based forwarding scheme. More precisely, when a packet is generated by the source node, it receives an amount of credits that it can spend during the forwarding process. The assigned credits are equal to the sum of the minimum cost from the source to the destination (i.e., the shortest path cost), plus extra credits necessary to expand the path while being forwarded. Then, whenever a node receives a packet, it decides if it is an appropriate forwarder according to the remaining credits of the packet and the cost of the shortest path from itself to the destination. To some extent, this extra credits reflects the level of resiliency demanded to the forwarding process. This strategy permits to create a forwarding mesh on-the-fly centered around the minimum-cost path from the source node to the destination node. Conceptually, this ap-

proach is similar to the one adopted in SOAR, which uses the ETX metric to control the width of the forwarding mesh, while ROMER credit mechanism provides a finer control on a per-packet basis. With this approach, a critical aspect is how to distribute packet credits along multiple candidate forwarders. For instance, more credits can be assigned to nodes closer to the source than to the destination. This has the effect of permitting a faster initial expansion of the forwarding mesh, and to remain closer to the shortest path when approaching the destination. However, other strategies taking into account interference or load distributions are equally valid design choices. To select the optimal forwarders among the nodes that received a packet, as well as to reduce the number of duplicate transmissions, ROMER employs a *probabilistic* strategy: the forwarding probability is set proportional to the link's current transmission rate and to the desired level of packet redundancy, so as to assign a higher forwarding probability to intermediate nodes that use higher transmission rates. This probability-based forwarding scheme permits to exploit the best-rate links that are dynamically identified by the rate adaptation algorithms (ABC08), but it is also beneficial to improve the routing resiliency to randomized packet losses, and to quickly adapt to the varying conditions of the wireless links.

A different approach for minimizing the coordination overhead is proposed in (ZKR08), which presents the *Transmit Diversity based Cooperative Opportunistic Routing* scheme (*TDiCOR*). Basically, TDiCOR employs passive listening for the forwarder selection: all the selected candidate forwarders that successfully received a packet try to forward it by contending simultaneously for the medium access. Then, the first candidate relay that gains access to the channel assumes the *forwarding responsibility*, which is the responsibility of continuing the packet forwarding to the destination, and retransmitting the packet if necessary. The other candidate forwarders overhearing this transmission will cancel their own. Thus, forwarder selection does not rely on any form of prioritization or cost-based scheduling, but it leverages only on random medium access. However, since the forwarders selection requires packet overhearing among candidate nodes, it is important that potential for-

warders have high-quality links between each other. To this end, TDiCOR uses the ETX metric to evaluate the closeness between nodes, which is somehow similar to SOAR (RSMQ06). Finally, to improve the reliability of frame transmissions, TDiCOR exploits the *transmit diversity* property. Specifically, after packet reception all the selected candidate forwarders transmit their link-layer ACK frames simultaneously, so that the sender receives multiple identical copies of the same ACK frame. This cooperative acknowledgment has the effect of increasing received signal strength and mitigating fading effects, while the next hop selection comes automatically from the medium access protocol without requiring any additional control traffic nor complex scheduling. However, the feasibility of transmission diversity depends on the level of timing accuracy. Finally, in TDiCOR transmit diversity is used not only for cooperative acknowledgements but also for cooperative data transmissions. Specifically, if the RTS/CTS option is active, when a node overhears a CTS frame it can check its interface queue to search if it has an identical frame to transmit (i.e., identical source address, sequence number, candidate set, and retry bit). In this case, that node can act as cooperative relay and it can transmit the frame simultaneously with the node that has sent the initial RTS frame.

2.3.2 Coding-based routing

As reported in Figure 5, coding-based routing solutions are divided into two sub-categories: *inter-flow coding* approaches and *intra-flow coding* approaches. The following two sections outline the most relevant schemes proposed for both approaches.

Inter-flow coding schemes

A fundamental approach proposed for inter-flow network coding in mesh networks is the *COPE* protocol (KRH+08). COPE relies on a legacy routing protocol to select a minimum-cost path (according to some metric) between nodes. In this sense, COPE is not an opportunistic routing protocol as defined in this study because the sequence of next hops that

each packet, encoded or not, should follow is fixed and known a priori. However, COPE also allows intermediate forwarders to mix packets from multiple unicast flows. To this end, each node implements opportunistic listening to overhear packets that are not intended to it but that can be used for efficient coding. The overheard packets are stored in a local buffer for a limited time period. Then, whenever the MAC protocol grants a node the permission to transmit, this node selects from its local buffers the packets to code together in such a way that all next hops of encoded packets will be able to reconstruct their corresponding native packets. More precisely, each node combines, using the *XOR* operator, n distinct packets headed for n different next hop relays only if it is sure that every intended next hop has already all the $n-1$ packets required to decode the native packets encoded together.

It is now clear that the critical aspect of the COPE design is to ensure that each node can learn the state of neighbors' buffers to know which packets they have. In practice, neighbors' buffer information is obtained through neighbors' notifications and "guessing". Specifically, each node broadcasts *reception reports* listing the packets it has stored. To some extent, reception reports are equivalent to the batch maps used in ExOR. The problem with the reception reports is that these messages can be lost or, even more important, arrive too late for the coding purposes. For these reasons, each node may anticipate if a particular packet has been received by a certain neighbor based on the delivery probability between that neighbor and the packet previous hop, i.e. the node from which it has received that packet. Furthermore, the packet coding algorithm in COPE is based on the principle of never delaying packets whenever the wireless medium is available. Thus, the node transmits a combination of packets if a coding opportunity exists, giving preference to packets of the same length, otherwise it simply forwards the native packet, if any, at the head of its transmission queue.

An interesting design choice of COPE is the use of *pseudo-broadcast* transmissions instead of conventional broadcast. More precisely, the destination MAC address of the encoded packet is set to one of the intended next hops, while an additional COPE-header specifies all the next

hops of the native packets mixed together. By setting radio interfaces in promiscuous mode, COPE enables each node to overhear multiple encoded packets, while, at the same time, unicast transmissions ensure a higher level of reliability. To further increase the communication reliability, a local recovery strategy is performed through hop-by-hop ACKs sent asynchronously by all the next hops to which the encoded packets were headed. If any of the encoded native packets was not acknowledged within a given timeout, the packet is retransmitted, possible encoded again, but with a different set of native packets. Note that decoding and re-encoding packets at each intermediate node is important to avoid diverging paths, resulting into packets that move away from their destinations. Indeed, this problem is originated by the use of inter-flow coding that allow to mix together packets even if they are headed for different areas in the network.

Experimental results shown in (KRH+08) indicate that the coding gain provided by COPE is highly dependent on a set of factors, including traffic patterns and congestion levels. To quantify these interdependencies in arbitrary topologies, authors in (SRB07) elaborate a linear programming formulation to model the maximum throughput achievable by COPE-style coding. An interesting contribution of this paper is also the notion of *coding-aware* and interference-aware routing for selecting routes that maximizes the coding gain while minimizing the interference due to the coded transmissions. However, the scheme proposed in (SRB07) requires an exhaustive analysis of all possible coding opportunities that may arise after a given routing decision. A simpler and more practical coding-aware routing approach, called *ROCX*, is described in (NSZN06). The ROCX scheme is based on a new routing metric, called *ECX*, which captures the expected number of coded transmissions needed to successfully deliver packets between two nodes communicating through a relay. Then, a linear programming problem is formulated to find paths between node pairs with the minimum ECX cost, i.e., which minimizes the expected total number of coded packets for a successful exchange of packets.

A common feature of the schemes described above is that the coding

process tries to minimize the number of coded transmissions needed to successfully delivery a set of packets. A different approach is adopted in (RSW+08), which proposes a series of algorithms for *network coding with loss-awareness* (*CLONE*), based on the idea that coding must provide higher levels of redundancy in lossy wireless environments. In other words, coding decisions aim to introduce an adequate redundancy in network coding operations in order to achieve higher reliability. Specifically, in (RSW+08) the binary (i.e., involving at most two native packets) network coding problem is modeled through a graph-based formulation. Then, various coding strategies are defined by imposing different constraints on the coding process. For instance, the CLONE-MultiXOR heuristic tries to maximize the number of ways a native packet can be decoded by the intended next hop. However, the high complexity required for packet selection limits the applicability of these algorithms beyond binary coding.

A somehow simpler approach to implement inter-flow network coding is proposed in (QXG+08) with the *Intra-flow&Inter-flow MIXing* protocol (*I*^{2}*MIX*). The basic idea behind I^{2}MIX is that each node with packets buffered in its transmission queue creates random linear combinations of the same subset of the stored packets until all the respective next hops acknowledge their correct reception. Intuitively, each receiving node that is a next hop sends an ACK as soon as it is able to recover the original data from the coded packets it has received. Then, the decoded packets are stored in the receiver's transmission buffer and are used to generated new coded packets. In contrast to COPE, I^{2}MIX generates random linear combinations of stored packets. This permits to take advantage of any existing coding opportunity, thus simplifying the coding process and avoiding the use of reception reports. However, in I^{2}MIX the sender can stop to send combinations of the same set of packets, and to move to the next one, only if it receives an acknowledgment from the next hop of each flow. In addition, the coding decisions cannot be optimized because the status of neighbors' buffers is unknown. Both these simplifications of the coding process can easily produce a number of transmissions much higher than the one that would be generally needed by COPE to deliver

the same number of native packets.

Intra-flow coding schemes

In principle, intra-flow coding would permit to avoid the problem of discovering the state of neighbors' buffers. However, coding packets flowing between the same source and destination pair could give rise to a limited number of coding opportunities. To address this issue, *Multipath Code Casting* protocol (*MC*^{2}) (GHK$^+$07) proposes to integrate coding with multi-path routing. Specifically, MC^2 relies on a legacy routing protocol to find a set of multiple, not necessarily disjoint, paths between a source and a destination node. This should ensure that multiple next hops exist for each node on a path. Then, relay nodes broadcast encoded packets generated using either native packets received by the source or other encoded packets received by neighbor nodes. Intermediate decoding is not allowed, and only the destination node collects encoded packets and reconstructs the original packets when it has received a sufficient number of linearly independent coded packets. Note that, similarly to ExOR, MC^2 performs its coding decisions on block of packets, also called *generations*, for limiting decoding overheads and state size at intermediate nodes. Finally, in order to provide reliability, two error control mechanisms are defined. First, a hop-by-hop local recovery is performed by each sending node, which overhears its neighbors' transmissions and retransmits some missed packets if necessary. Second, on a timeout the destination sends a unicast request for additional coded packets to the source, which can also be intercepted and managed by any intermediate node holding the missed packets.

The critical part of the MC^2 scheme is how to assign coding rates to the multiple paths, i.e., how to decide which next hop a packet must be sent to, and how many encoded packets should be generated along each path. To this end, a credit-based algorithm is proposed in (GHK$^+$07). Specifically, the source associates to each packet generation a given amount of *credits*, which represents the total number of packets (i.e., including coded packets) that should be used to transfer that block of native packets to the destination. Moreover, the number of packets the

source is allowed to send per time unit is specified as a function of the generated credits. Then, whenever a packet is successfully transmitted to a next hop, a credit is also transferred to that node. Based on the credits associated to each node, forwarders are aware of the amount of packets that are waiting to be sent over each link. Then, to determine the best next hop to which a node should forward the packet it has received, the routing protocol applies the back-pressure algorithm (TE92) to the total credits accumulated by each node. In other words, the packet transmission is scheduled on the broadcast link with the maximum difference in the queued credits. The drawback of this approach is that this optimal scheduling is hard to implement, and the required node state grows exponentially with the number of neighbors.

An alternative approach to implement intra-flow coding by exploiting node overhearing is proposed in the *Intra-flow MIXing* (*IMIX*) protocol (QXG+08). The basic principles are similar to the ones used in I^{2}MIX: the sender keeps coding the same set of n packets from the same data flow with random linear coding until receiving an acknowledgement from the next hop. Obviously, the next hop can generate an acknowledgment only after having received n linearly independent coded packets. Then, the final destination will recover the native data packets from the coded packet, while intermediate nodes only recode the received packets. It is important to note that with traditional acknowledgments, one ACK message is needed per every received data packets. On the contrary, with intra-flow coding only one acknowledgement is needed for every n packets. Thus, using linear coding it is possible to reduce the overhead of the acknowledgment process without the cost of increased protocol complexity. To some extent, IMIX can be viewed as a basic intra-flow solution in which every node simply codes and broadcasts packets stored in its buffer, irrespective of those packets received by its neighbors. To maximize the coding gain, which can be low for intra-flow coding since coding opportunities may be scarce, IMIX employs a coding-aware routing protocol, called *OSPR*, which selects network paths with least ETX value, taking into account overhearing opportunities. Compared to classical shortest path routing, OSPR network paths generally

include more hops to provide more overhearing opportunities.

2.3.3 Hybrid routing

As reported in Figure 5, hybrid routing solutions are divided into two sub-categories: *destination-based* approaches and *neighborhood-based* approaches. The following two sections outline the most relevant schemes proposed for both approaches.

Destination-based schemes

The *MAC-independent Opportunistic Routing and Encoding* protocol (*MORE*) proposed in (CJKK07) is the first practical system that combine intra-flow random linear coding with ExOR-style opportunistic routing. As in MC^2 (see Section 2.3.2) packets are grouped into blocks, which are now called *batches*, and coding is restricted to linear combinations of packets of the same batch. Similarly to ExOR (BM05), opportunistic routing is performed by letting the sender specify a prioritized list of candidate forwarders. However, in contrast to ExOR there is not a structured transmission schedule among the forwarders, but the coding permits random transmissions regulated through the 802.11 MAC protocol. More precisely, the source breaks up the file to be transmitted into batches of native packets, creates random linear combinations and broadcasts the resulting packets after adding a MORE header containing the forwarder list. Each receiving node discards a packet if it is not innovative, i.e., not linearly independent from the other packets stored in the node's local buffer, or if the node does not appear in the associated forwarder list. Otherwise, it refreshes the packet stream by linearly combining the received coded packets and rebroadcasting the newly encoded packets. Note that a linear combination of coded packets is also a linear combination of the corresponding native packets (CJKK07). As soon as the destination is able to decode the whole batch, it sends an ACK to the source using shortest path routing, causing the sender to stop forwarding packets from that batch and start processing the next batch. The intermediate nodes stop coding/sending packets from a certain batch as soon as

they intercept the ACK for that batch sent by the destination, or they receive a packet belonging to a new batch. This strategy leads to a faster synchronization among nodes without requiring complex coordination procedures.

Although network coding reduces the number of required transmissions for successfully delivering a packet, the opportunistic paradigm allows any potential forwarder to send many coded packets. However, uncontrolled generation of coded packets results in redundant transmissions. The trade-off is between transmitting a sufficient number of coded packets to guarantee that the destination has enough innovative packets to reconstruct the native packets, and avoiding to inject in the network unnecessary packets that may cause congestion. To address this issue, MORE uses a heuristic algorithm to estimate the maximum number of transmissions that each node can perform after receiving a packet from an upstream node, which is a node farther from the destination than itself. This limit is computed by each node considering the loss probability in sending a packet to its neighbors, and the probability that the packet to be transmitted has not been yet overheard by downstream nodes, which are nodes closer to the destination than itself. Experimental results obtained in an indoor wireless testbed indicate that MORE's throughput gain over ExOR can be relevant when there is a chance of spatial reuse, because MORE allows multiple forwarders to access the channel simultaneously, which is hindered in ExOR (CJKK07).

As pointed out above, determining the acceptable rate for intermediate nodes represents a critical issue for hybrid schemes, especially in scenarios involving multiple flows. Thus, a number of recent papers have proposed optimization approaches for broadcast rate control, which maximize the benefit of network coding and broadcast transmissions while mitigating congestion. For instance, the *Optimized Multipath Network Coding* scheme (*OMNC*) proposed in (ZL08b) formulates the throughput-maximization problem as a linear programming problem, whose outcome is the optimal encoding and broadcasting rate for all nodes. A variant of this approach, called *DICE*, is proposed in (ZL08a) by taking a game-theoretic perspective. However, these enhancements

of MORE require the exchange of a large amount of state information, which may be quite inefficient in lossy environments. A different and more practical scheme to improve routing efficiency is the *CodeOR* protocol proposed in (LZL08). The design idea of CodeOR originates from the observation that the coding process in MORE, and subsequent enhancements, is similar to a 'stop-and-wait' protocol because the source keeps coding packets of the same block, also called *segment* in CodeOR's notation, until it receives an explicit signal from the destination. In contrast, CodeOR allows the source to transmit a *sliding window* of multiple segments so that data coding/broadcasting is not limited to only one block of packets. Moreover, each intermediate node locally decides when it should start processing a new segment within the allowed window. Hence, we can say that coding decisions depend globally on the destination, but may adapt locally to the current situation. In practice, this adaptive behavior is implemented using two different ACK messages: an end-to-end ACK (called E-ACK) sent by the destination directly to the source node, and a hop-by-hop ACK (called H-ACK) sent by intermediate nodes. The former message is used to indicate that a segment of data packets have been received at the destination, and it regulates the window-based flow control similarly to TCP ACKs. The latter message is generated by a node to inform its upstream nodes that it has received a sufficient number of coded packets of the current segment so that it can continue the coding process on behalf of the source node, while the upstream nodes can move to the next segment. The peculiarity of the hop-by-hop support proposed in CodeOR is that the number of required coded packets is not constant but depends on the receiving rate of each node. Specifically, each node computes a *receiving threshold* proportional to its receiving rate, which specifies when it can assume that it has received enough packets in a segment. This rate-based threshold aims to balance the number of packets generated for a certain segment with the reception and decoding rate at the various downstream nodes.

Neighborhood-based schemes

All the previous hybrid schemes adopt intra-flow coding because with opportunistic routing each packet has more than one possible next hop, and coordinating the coding process through multiple flows is probably less intuitive. In contrast, the hybrid scheme proposed in (KHW08), called *XCOR*, is an attempt to combine inter-flow coding with opportunistic routing. To achieve this goal, XCOR abandons the approach followed in destination-driven coding strategies (e.g., MORE (CJKK07) and, partially, CodeOR (LZL08)), to adopt a COPE-style coding, where the coding decisions are driven by neighbors' notifications and overhearing of neighbors' transmissions. More precisely, in XCOR the source constructs the set of nodes allowed to participate in the forwarding process of a certain packet starting from the shortest path, and sorting the candidate next hops in terms of ETX-proximity to the destination. Then, after each transmission, the nodes on the shortest path are allowed to rebroadcast immediately the packets they receive, while the other relays set a forwarding timer in proportion to their priority. In this way, if they overhear a transmission from a higher priority node, they can cancel their timers. Furthermore, similarly to COPE (KRH$^+$08) each node periodically sends reception reports to inform its neighbors about the packets it has received. The most innovative aspect of XCOR scheme is the way these reports are used to regulate the mixing of packets. Specifically, let assume that a node is crossed by m different flows. Then, the node computes the utility of each possible combination of packets belonging to these m flows , in order to find the one that gives the largest utility. However, the number of possible combinations increases exponentially with the number of flows to code together. For this reason, XCOR applies an heuristic that examines the flows in a sequential order, giving higher priority to flows that are heavily loaded, so that the packet dropping probability is minimized.

2.4 Discussion

In this chapter, we have examined the key challenges associated to the design of routing algorithms that use opportunistic forwarding and network coding to take advantage of the multi-user diversity and the broadcast nature of the wireless medium. To this end, we have presented a taxonomy of existing solutions relying on these novel routing paradigms, and we have analyzed their most representative features, relative strengths and weaknesses. From this overview, it is easy to identify some common functionalities and mechanisms that can be considered as basic building blocks for each solution. Thus, in the following we integrate the previous discussion by summarizing in Table 1 the specific design choices made by the presented solutions for each of these key components. The aim of this schematic illustration is to further clarify the main features of the various schemes, as well as to permit an easier comparison between the various approaches. To this end, the first column specifies the solution name, as reported in the proposed taxonomy (see Figure 5), while the other columns are dedicated to the most representative features of diversity-based routing approaches, which are a subset of the key challenges illustrated and discussed for each category. Hence, by looking at this concise description, basic differences and common aspects among the various approaches are immediately noticeable. In the following we briefly explain the meaning of the columns fields:

Routing: This field specifies if the routing approach is opportunistic or legacy. The former implies broadcast transmissions at each hop, while the latter can perform either unicast or broadcast/pseudo-broadcast transmissions, as explained in Section 2.3.2. Moreover, legacy routing can be either single-path or multi-path.

Scheduling: This field specifies if candidate forwarders coordinate their transmissions by establishing an ordering among them, or if each node autonomously decides whether to carry on with the forwarding process. In legacy routing approaches, a node can transmit only after receiving a packet from the previous hop along the predeter-

mined path, thus the scheduling is not a task to be performed at the routing layer. On the other hand, this aspect is particularly important in the opportunistic routing case, where different solutions have been proposed.

Node priority: This field specifies the metric used for establishing node ordering. Some schemes propose new metrics, such as ECX (NSZN06), or use existing ones, such as ETX. Other schemes rely on path costs for routing operations, but they are not forced to use a specific routing metric. It is important to note that node priority is not necessarily associated to scheduling. For instance, in some cases node priority is used to prune some nodes from the forwarding process, e.g. if the ETX of a node is greater than that of the sending node. Finally, in legacy routing node priority is in general provided implicitly by the predetermined sequence of next hops in the selected network path(s).

Coding: This field reports, when applicable, the coding technique and the coding strategy. Generally, the coding techniques used in the presented schemes can be either the *XOR* operation or the random linear coding (*RLC*). On the other hand, the coding strategy refers to the possibility of combining packets belonging only to the same data flow (intra-flow coding) or to different flows (inter-flow coding).

Duplicate suppression: This field specifies the method used for minimizing the duplicate transmissions. This issue is particularly relevant for opportunistic based routing, since many potential forwarders may send the same packets, leading to unnecessary transmissions. In case of network coding applied to legacy routing, duplicate suppression is not required because the routing protocol allows only some predetermined nodes to send data, each one towards its respective next hop(s). In principle, the redundancy provided by network coding is intended to favor packets decoding, thus duplicate suppression is not an issue. In contrast, opportunis-

tic forwarding suffers from duplicate transmissions because there is a looser control on packets dissemination.

ACK strategy: This field indicates the method used to acknowledge packet transmissions. Most of the network coding based solutions rely on link-layer ACKs, while hybrid approaches mainly use end-to-end ACKs, although some hybrid schemes may also use some kind of local recovery mechanism. The opportunistic-based routing schemes offer a higher variety of solutions, ranging from hop-by-hop network-layer ACKs to priority-based link-layer ACKs, and batch maps. In a few cases, the acknowledgment approach is not specified, thus it is not reported in the table.

Prototype: This field states whether a real implementation of the solution exists, "no" meaning that the protocol evaluation has been carried out based only on simulations.

Protocol	Routing	Scheduling	Node priority	Coding	Duplicate suppression	ACK strategy	Prototype
ExOR	opportunistic	yes	ETX-based distance to destination	no	batch map and priority-based forwarding timers	batch map	MIT Roofnet testbed
MCExOR	opportunistic	yes	ETX-based distance to destination and home channel	no	priority-based link-layer ACKs	hop-by-hop slotted ACKs	no
SOAR	opportunistic	yes	ETX-based distance to destination + ETX-based proximity to minimum-cost path	no	overhearing and priority-based forwarding timers	hop-by-hop network-layer selective ACKs	no
OpRENU	opportunistic	yes	residual network utility	no	priority-based link-layer ACKs	link-layer ACKs	no
ROMER	opportunistic	no	minimum-cost path to destination + link data rate	no	overhearing and randomized forwarding	-	no
TDiCOR	opportunistic	no	ETX-based distance to destination and other candidate forwarders	no	overhearing	cooperative acknowledgement	no
COPE	legacy with pseudo-broadcast transmissions	-	-	inter-flow XOR with reception reports/guessing	-	link-layer ACKs	20-node indoor testbed
ROCX	legacy and coding-aware	-	-	inter-flow RLC	-	-	no
CLONE XOR	legacy with broadcast transmissions	-	-	inter-flow loss-aware XOR with reception reports/guessing	-	link-layer ACKs	12-node testbed
I^2MIX	legacy with broadcast transmissions	-	-	inter-flow RLC	-	link-layer ACKs	no
MC^2	legacy multipath with broadcast transmissions	yes	transmission credits	intra-flow RLC	-	hop-by-hop overhearing, end-to-end retransmissions	MIT Roofnet testbed
IMIX	legacy and coding-aware	-	-	intra-flow RLC	-	MAC-layer unicast ACKs	no
MORE	opportunistic	no	ETX-based distance to destination	intra-flow RLC	packet innovativeness + overhearing of a new batch + ACK overhearing	end-to-end ACKs	MIT Roofnet testbed
OMNC	opportunistic	no	ETX-based distance to destination	intra-flow RLC	packet innovativeness + overhearing of a new generation + ACK overhearing	end-to-end ACKs	emulation testbed
CodeOR	opportunistic	no	ETX-based distance to destination	intra-flow RLC	packet innovativeness + hop-by-hop ACKs	end-to-end ACKs	no
XCOR	opportunistic	yes	ETX-based distance to destination + ETX-based proximity to minimum-cost path	inter-flow XOR with reception reports	overhearing and priority-based forwarding timers	reception reports	no

Table 1: Summary of the key design choices of the wireless diversity based routing approaches presented in this chapter

Chapter 3

Novel Opportunistic Routing Paradigms

3.1 Background and Motivation

In the previous chapter, we have provided an extensive overview of routing approaches that rely on wireless diversity to overcome performance degradation in WMNs. In this context, we consider *opportunistic routing* as a fundamental building block of any routing solution for WMNs. Motivated by its benefits and stimulated by the necessity to overcome limitations of existing solutions, we firstly focus on this category and we propose novel algorithms aimed to overcome the limitations of existing approaches to further improve network performance. Indeed, as described in the previous chapter, to limit the coordination overhead among possible packet forwarders, many existing opportunistic routing protocols (BM05; RSMQ09) select a priori a small list of candidate forwarders, generally prioritized by closeness to the destination, and only those nodes can be used to reach the destination. Thus, the fundamental limitation of this approach is that candidate forwarders are *pre-selected* before the packet is received. Many variants of this basic approach have been proposed, while a very few schemes try to avoid the pre-computed forwarding scheduling. ROMER (YYW+05) employs a credit-based for-

warding scheme to construct at runtime a mesh of forwarders centered around the minimum-cost path. However, forwarders selection is driven by the packet credit assignment performed at the source node, thus imposing a priori limitations on the routes a packet is allowed to follow. On the other hand, TDiCOR (ZKR08) simplifies the forwarder selection by fully relying on the random medium access mechanism: the next hop of a packet is the first receiver that gains access to the medium among the candidate relays specified in a forwarders list. Although the simplicity of this solution reduces node coordination overhead, packet forwarding depends only on random channel access and can not be driven by any specific goal (e.g. selecting nodes closest to the destination to achieve higher throughput).

Motivated by the above considerations, in this chapter we present two innovative opportunistic routing algorithms, aimed to improve the performance of WMNs by avoiding a priori limitations on candidate forwarders and on admissible routes leading to the destination. Indeed, they perform routing decisions only after packet reception and maintain node coordination overhead limited. In the following sections, we explain in more details the proposed algorithms, and we provide experimental results to show their effectiveness.

3.2 Maximizing Throughput Gain with Opportunistic Routing: MaxOPP

In this section we present MaxOPP, a novel opportunistic routing algorithm for WMNs, which is based on a substantially different approach than existing solutions. Specifically, MaxOPP abandons the precomputation of candidate forwarders, and it does not force selected forwarders to transmit during pre-assigned time windows. On the contrary, wireless diversity generates multiple receivers for each packet transmission, and any of those receivers could be used as an alternative forwarder to reach the final packet destination. Thus, MaxOPP adopts a more flexible and resilient approach by employing a localized routing decision process that selects the forwarding nodes at runtime and on a per-packet

basis. In this way, MaxOPP can opportunistically leverage any transmission opportunity generated by the short-term channel dynamics, and limit the probability of excluding beneficial forwarders. Moreover, MaxOPP bases the forwarder selection process on an estimate of the opportunistic throughput gain associated to the packet transmission, allowing each packet to flow through the most advantageous forwarder at each hop.

3.2.1 MaxOPP Design

In this section, we firstly present the main principles of MaxOPP design, and then we describe the MaxOPP algorithm in details.

Overview

To clarify how the MaxOPP scheme works, we can start by observing that when a packet traverses a route it consumes network resources. For this reason, many routing metrics exist to estimate the forwarding cost associated to each network path. Traditional routing algorithms search for minimum cost paths that ensure a long-term stable optimality of some performance metric (e.g., hop count, or average path throughput). On the contrary, our opportunistic routing solution adjust at runtime, and on a per-hop basis, the route followed by a packet to ensure improved opportunistic throughput gain and higher reliability. More precisely, upon receiving a broadcast transmission, a node checks if further forwarding the packet can minimize the expected cost to reach the packet destination. Intuitively, if the packet has travelled along long links, consuming significantly less resources than the ones demanded by the minimum-cost path, continuing to forward the packet may provide an opportunistic throughput gain. Thus, before deciding whether to continue to forward a packet or not, the receiver checks the network resources consumed so far by the received packet, as well as the remaining path cost to reach the destination. By completely deferring the forwarder selection after packet receptions, MaxOPP is able to opportunistically adapt the forwarding process to the dynamic channel conditions, limiting the

probability of excluding beneficial forwarders.

In order to realize the opportunistic scheme described above, it is necessary to address two fundamental issues. The first one is how to compute at run time the potential throughput gain associated to a packet transmission. The second one is how to control the overhead due to the delivery of redundant copies of a packet. In the following sections we describe how MaxOPP solve those issues.

MaxOPP forwarding procedure

As in most of the existing opportunistic solutions, MaxOPP assumes that an underlying link state routing protocol is responsible for collecting link qualities and disseminating this information to all the nodes in the network. Thus, any node has a complete knowledge of the network topology and can pre-compute the "best" path to reach any other node in the network according to some routing metric. Note that an important property that a routing metric should satisfy is *isotonicity*, since this property determines if efficient algorithms such as Dijkstra or Bellman-Ford can be used to find minimum-cost paths, and whether hop-by-hop routing protocols yield loop-free paths (YWK05). In the following evaluation we will use ETX (DCABM03) as the underlying routing metric, but any alternative isotonic routing metric can be used.

Before describing the MaxOPP forwarding procedure, let us introduce some useful notation. First of all, let us denote with $cost(p)$ the cost of delivering a data packet on path p according to the chosen routing metric. Then, let $p_{S,D}$ be the Minimum Cost Path (MCP) from node S to node D. Thus, the cost of delivering a packet on the MCP from node S to node D is simply $cost(p_{S,D})$. Now, let us assume that a packet generated by node S for node D, and forwarded on the MCP has traversed k hops along $p_{S,D}$, and let i be the k-th intermediate router on $p_{S,D}$ that has received this packet. For brevity, we define $i = p_{S,D}(k)$. Then, we express the cost of the remaining portion of the MCP from S to D after k hops as $cost(p_{S,D}; k)$. From the isotonicity property of the routing metric, it immediately follows that $cost(p_{S,D}; k) = cost(p_{i,D})$. In the following, we show how MaxOPP exploits the per-packet knowledge of the

$cost(p_{S,D}; k)$ value to estimate the benefit of using an intermediate node as the next packet forwarder.

In MaxOPP, all packets are broadcasted and keep track of the number of times they have been forwarded by intermediate nodes[1]. Now, let us suppose that a node j hears a packet transmission belonging to a traffic flow from S to D, and which has travelled along k wireless hops. Then, based on the above introduced notation, $cost(p_{j,D})$ is the minimum cost path from node j to node D. On the other hand, if the packet had been routed along the shortest path between S and D (i.e., $p_{S,D}$), the remaining cost after k hops would had been equal to $cost(p_{S,D}; k)$. Hence, we can evaluate the potential benefit of using j as the next forwarder by introducing an *opportunistic gain* as follows

$$OG_{SD}(j, k) = cost(p_{S,D}; k) - cost(p_{j,D}) \ . \tag{3.1}$$

The above value represents a sort of *dynamic credits* that would be granted to the packet if node j is used as the k-th forwarder for the traffic flowing from S to D. Clearly, transmissions from nodes along the minimum-cost path provide no gain, given that $cost(p_{S,D}; k) = cost(p_{i,D})$, where $i = p_{S,D}(k)$. If the opportunistic gain is negative, this implies that the packet is lagging behind the minimum-cost path instead of keeping up with it, and it should be discarded by node j as its further transmission would not be beneficial.

The fact that $OG_{SD}(j, k) > 0$ is a necessary but not sufficient condition to select node j as potential forwarder. Indeed, MaxOPP is designed to guarantee that each transmission provides a minimum level of opportunistic gain, avoiding transmissions that would decrease the opportunistic benefit achieved so far. Thus, the forwarding decision at node j should depend also on the gain obtained in the previous hops. More precisely, let us assume that a node l, upon receiving a packet belonging to a traffic flow from node S to node D, and which has traversed $k-1$ hops, decides to further forward it. If the new packet transmission is received

[1]To this end, we assume that a tiny MaxOPP header follows the MAC-level header and precedes the packet's data. This MaxOPP header is used to carry control information that are needed for executing the MaxOPP forwarding procedure.

by node j, this node can compute the *opportunistic gain ratio* as follows:

$$OR_{SD}(l, j, k) = \frac{OG_{SD}(j, k)}{OG_{SD}(l, k-1)} \ . \tag{3.2}$$

Intuitively, a gain ratio greater than one would mean that the opportunistic gain is increasing from one hop to the subsequent, and that the forwarded packet is traveling towards the destination on a shorter path than the long-term minimum cost path. However, MaxOPP aims at improving not only throughput performance but also resiliency to lossy links or transient node failures. Therefore, the MaxOPP forwarding procedure should ensure that a sufficient number of potential forwarders is activated, in order to increase the probability that at least one of the packet copies is received correctly by neighboring nodes. In other words, a high threshold on the $OR_{SD}(l, j, k)$ value could be an obstacle for the forwarding process. More formally, in MaxOPP a node j is allowed to forward a packet flowing from S to D and received after k hops from node l if and only if

$$OR_{SD}(l, j, k) \geq \gamma \ , \tag{3.3}$$

where $\gamma \geq 0$. The choice of the γ value provides enough flexibility to support the desired level of resiliency under different network scenarios or traffic demands. For instance, the γ parameter can be a function of the distance between the source and the destination, or of the desired bound on the total number of generated packet copies, or it can be adaptively adjusted during packet forwarding depending on the channel conditions. In the following evaluation, we set $\gamma = 0.8$, but we tried also other values without observing significant performance differences.

Note that a node may receive multiple copies of the same packet. To avoid unnecessary replicated transmissions, each node stores locally the sequence numbers of its recently forwarded packets. Upon receiving a packet, the node checks if it is a duplicate, and in this case discards it. In this way, each node forwards the same packet at most once. Furthermore, the same packet can be received by multiple receivers and the MaxOPP forwarding procedure is performed independently on each of them. Thus, multiple nodes can be selected simultaneously as poten-

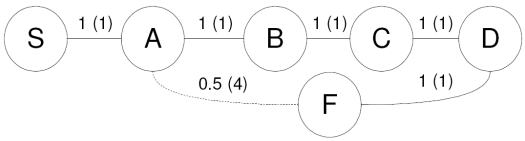


Figure 6: Example scenario: the label associated to each link is the delivery rate (links are symmetric), while in parenthesis we report the corresponding ETX cost computed according to formula in (DCABM03).

tial forwarder for the same packet. To minimize forwarding overhead, MaxOPP implements *overhearing* between nearby nodes. In other words, whenever a node receives a packet it also checks if it has the same packet in its transmission buffer waiting for being transmitted. In this case, it cancels its own transmission and removes the packet from its queue. In this way, the node that transmits first is the one that continues the packet forwarding. Note that overhearing is commonly adopted in existing opportunistic routing solutions as a low-overhead and distributed technique to coordinate forwarding nodes (BM05; RSMQ09; YYW$^+$05).

To conclude this section, we illustrate the MaxOPP algorithm using the example in Figure 6. In the drawing, S is the traffic source and D is the destination. A minimum cost path routing would select *S–A–B–C–D* as the best path, because the path cost is 4.0, while the other network path $S-A-F-D$, although shorter in terms of hops, has a higher cost equal to 6.0 due to the high packet loss associated to the link from A to F. However, let as assume that a packet sent by node A is received from both node B and node F. In this case $cost(F-D) = 1$, while $cost(B-C-D) = cost(p_{S,D}; 2) = 2$. According to formula (3.1), it holds that $OG_{SD}(F, 2) = 1$ and $OG_{SD}(A, 1) = 0$. Thus, from (3.2) it follows that $OR_{SD}(A, F, 2) \rightarrow \infty$, which is greater than any threshold γ. This implies that MaxOPP selects node F as potential forwarder for the received packet. If node F succeeds in transmitting the packet before node B, which depends on the dynamics of the MAC contention resolution scheme, the latter cancels its copy of the packet in order to keep data redundancy limited.

Loss recovery

Loss recovery is one of the key challenges of opportunistic routing design. Indeed, legacy 802.11 does not provide any link-level recovery mechanism for broadcast transmissions. For this reason, opportunistic routing protocols generally introduce network-layer acknowledgments and retransmissions, either of hop-by-hop or end-to-end basis. In MaxOPP we adopt an end-to-end acknowledgment scheme similar to the one used in (RSMQ09). More precisely, an end-to-end ACK message is periodically sent to the source by the destination along the shortest path using MAC-layer unicast. This acknowledgement message contains the sequence numbers of the lost packets, which are provided by means of a fixed-size bitmap to keep the overhead limited. In addition, the ACK packet contains some additional information that may help the source node in setting protocol parameters in a proper way. For example, the source node may decide to adjust the gain ratio threshold (i.e., the γ value) according to the number of lost packets announced by the destination. Note that the γ parameter is announced to the other nodes in the MaxOPP header. We point out that packet redundancy is also an indirect way of ensuring loss recovery. In fact, increasing the number of forwarded copies in a controlled manner may be useful to ensure that at least one packet copy is correctly received. It is important to note that data redundancy aims to protect against packet losses during the forwarding process itself, while end-to-end acknowledgments recover lost packets not received by the destination within a reasonable amount of time.

3.2.2 Performance Evaluation

In this section, we evaluate the performance of MaxOPP using NS-2 simulations in a set of representative network scenarios. Furthermore, we compare MaxOPP against OLSR, a widely adopted link-state single-path routing protocol that forwards packet over shortest paths. Our results show that MaxOPP significantly improves throughput of bulk transfer over traditional routing.

Simulation set-up

We implement MaxOPP in NS-2 (version 2.33). For comparison, we use OLSR-ETX code, which is an open-source implementation of the ETX-based OLSR protocol for NS-2 (W. 09). The ETX-based link cost is computed by measuring the number of OLSR control messages lost every 10 packets sent.

To be able to conduct experiments in controlled environments, while reproducing the behaviors of lossy wireless links, we extended the physical channel model of NS-2 to generate packet losses by dropping packet received at the MAC layer with a constant probability. Note that in our tests we have uses 802.11 MAC DCF scheme with fixed transmission rate equal to 11 Mbps, and disabled RTS/CTS. Concerning the traffic model, we use CBR UDP flows generating fixed-size packets at a rate sufficiently high to saturate the wireless channel. If not otherwise stated, the packet size is 1000 bytes. We have not considered interactive traffic such as TCP or VoIP because MaxOPP is designed to deliver bulk data faster than traditional routing. Better integration with interactive traffic will be the subject of future work. Each simulation run consists of 300-second data transfer, but throughput measurements are not collected during the first 150 seconds to let the ETX metrics converge. In order to collect statistics (i.e., average values and 95% confidence intervals) we replicate each simulation ten times.

Simulation results

We evaluate MaxOPP under a range of traffic demands and network topologies. Initially, we investigate the performance of a single flow in a basic chain topology, then we study multiple flows in more complex grid topologies.

Single Flow We use linear chain topologies with varying number of hops to evaluate the efficacy of MaxOPP to leverage transmissions that unexpectedly reach far nodes, while mitigating the negative impact of failed transmission attempts. Figure 7 exemplifies a 4-hop chain topol-

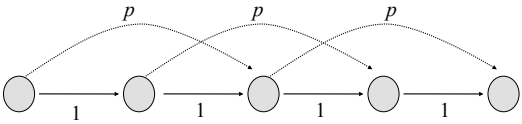


Figure 7: Illustration of a chain topology used for evaluation: the label associated to each link is the delivery rate (links are symmetric).

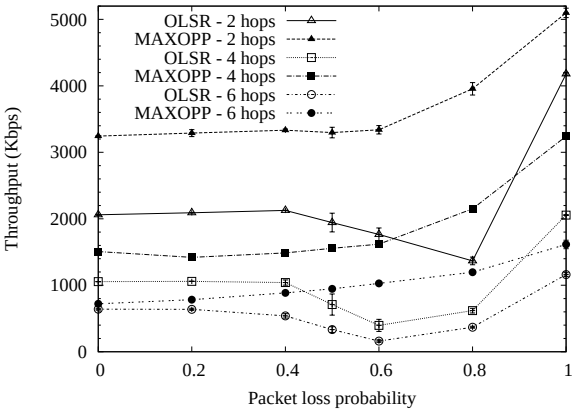


Figure 8: Throughput in a chain topology versus the packet loss rate.

ogy used in the simulations. Specifically, as shown in the diagram, all 1-hop links are noiseless, i.e., the delivery rate of those links is one and packets can get lost only due to collisions. On the contrary, 2-hop links have a fixed delivery rate p, which may be lower than one.

Figure 8 compares the per-flow throughput achieved with MaxOPP and ETX-based OLSR in the chain topology by varying both the number of hops between the source and the destination, and the packet delivery rate p. We can observe that MaxOPP performs significantly better than shortest path routing in all the considered scenarios, with a throughput gain that can be higher than 200% for short chains and packet loss rates from moderate to high values. More generally, the throughput gain is larger for packet delivery rates in the range from 0.4 to 0.8. This can

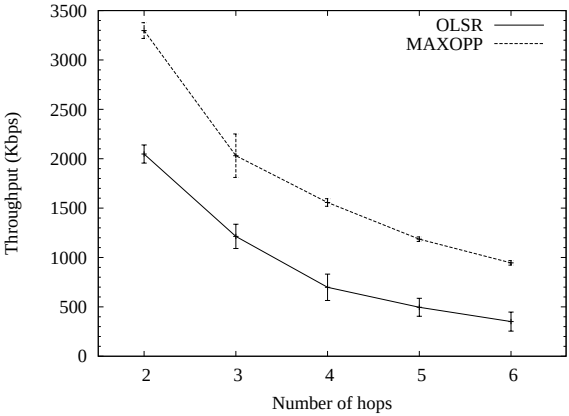


Figure 9: Throughput in a chain topology versus the number of hops.

be explained by observing that if the delivery rate is too low (i.e., delivery rate close to 0.0) than there a few opportunities to take advantage of long transmissions, while if the delivery rate is close to one, the shortest path routing already directly use network routes with less links. However, also in the extreme cases (i.e., delivery rate either equal to 0.0 or 1.0) MaxOPP may provide a positive throughput gain over shortest path routing. This can be explained by observing that MaxOPP reduces the MAC overheads because it eliminates hop-by-hop link-layer acknowledgements, and it is less affected by individual packet losses. Note that for intermediate delivery rate OLSR suffers from throughput degradation, as it shown in the Figure 8. This can be explained by observing that the variability of link quality estimation may cause frequent route flapping during the flow lifetime. These behaviours have been also observed in real network deployments, as reported in (RSBA07).

Finally, Figure 9 compares MaxOPP with ETX-based OLSR for linear topologies by varying the number of wireless hops between the source and the destination but fixing the packet delivery rate at 0.5. It can be observed that the throughput gain is almost independent of the number of hops.

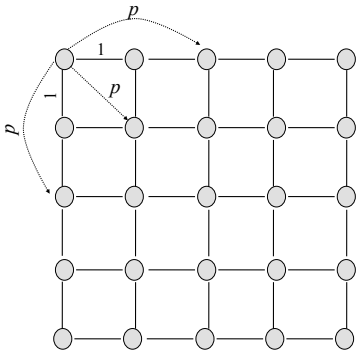


Figure 10: Illustration of the 5×5 grid network topology used for evaluation: the label associated to each link is the delivery rate (links are symmetric).

Multiple Flows To evaluate the performance of MaxOPP with multiple flows we consider a 5×5 grid-based network topology where delivery rates for 1-hop links are perfect, while delivery rates of 2-hop links are equal to 0.5, as shown in Figure 10. For each test we fix the number of flows in the network and we randomly pick up source and destination nodes. The only constraint we impose is that each node is either the source or the destination of a single flow. Intuitively, this limits the maximum number of flow that can be activated in a 5×5 grid topology to twelve connections.

Figure 11 and Figure 12 show the absolute aggregated throughput and percentage improvement, respectively, for different number of flows. The average percentage improvement is computed by calculating the ratio between the total throughput achieved by MaxOPP and OLSR for each run, and then evaluating the mean value. Note that while the total aggregated throughput is a statistic dominated by the traffic scenarios that ensure the largest values, the percentage improvement is calculated in such a way to assign the same weight to all the runs.

As shown in the pictures, MaxOPP outperforms the shortest path

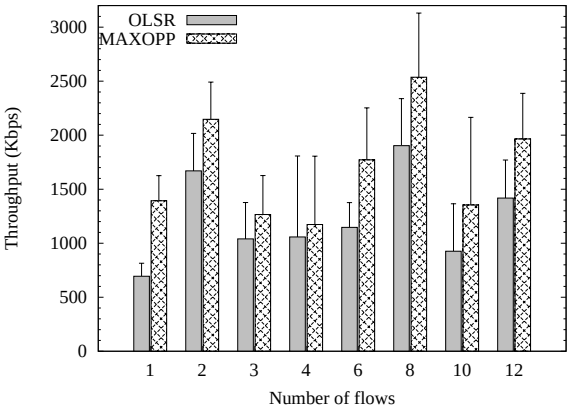


Figure 11: Throughput in a 5×5 grid topology versus the number of flows.

routing with an average improvement that ranges from 10% with 4 flows to more that 100% with one flow. Note that the confidence intervals of the results is much higher than in the case of a linear topology with a single flow. This can be explained by noting that in each test flows are selected randomly and there may be significant differences in the spatial distribution and lengths of flows in different runs.

3.3 Improving Efficiency in Multi-flow Scenarios: PacketOPP

As shown in the previous sections, MaxOPP exploits the notion of opportunistic throughput gain associated to each packet transmission to ensure higher throughput than conventional shortest path routing. It belongs to the sub-category of not-scheduled opportunistic routing protocols (2.3.1), which avoid strict node scheduling to leverage potentially any transmission opportunity encountered during packet forwarding. However, when the network congestion arises due to an increased number of flows, the efficiency of such schemes generally decreases because a packet is allowed to be forwarded by multiple nodes at each hop, which

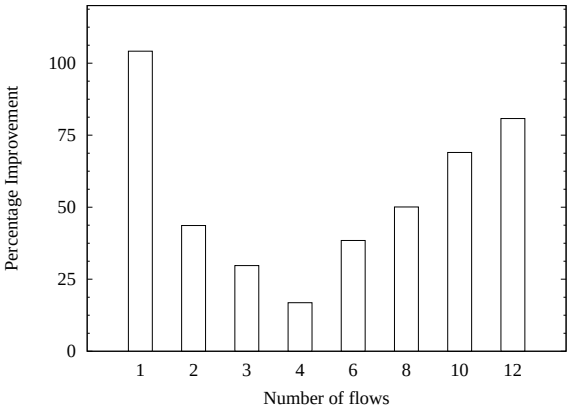


Figure 12: Throughput improvement in a 5×5 grid topology versus the number of flows.

may exacerbate congestion conditions. As a result, efficient support of multiple flows with opportunistic routing appears still an open problem.

To address the above limitations, in this section we present and evaluate PacketOPP, a novel opportunistic routing protocol that combines randomized opportunistic forwarding with *opportunistic packet scheduling*. The starting observation is that, if all packets are buffered in a single queue, and the forwarders transmit the received packets using a FIFO service discipline, this would make impossible to take advantage of the fact that some blocked packets in the queue can guarantee a higher potential throughput increase than the Head-of-Line (HOL) packets. To cope with this undesired restriction, PacketOPP employs packet scheduling techniques to award a higher priority to the packets that are expected to deliver the highest opportunistic gain, ensuring not only throughput gain maximization but also a more efficient duplicates suppression, limiting congestion increase.

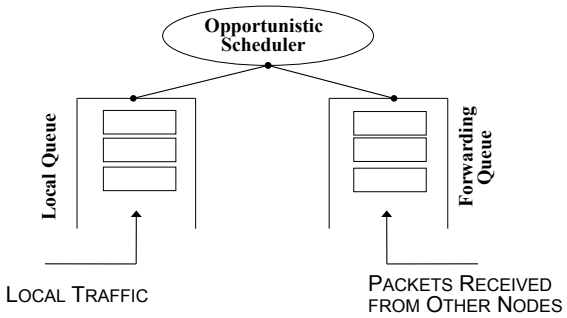


Figure 13: Node model.

3.3.1 PacketOPP Design

In this section we firstly present an overview of the design principles for the PacketOPP protocol, and then we describe each component of PacketOPP in details.

Overview

As a non-scheduled opportunistic routing protocol, PacketOPP does not employ a scheduler to regulate the order in which potential forwarders should transmit, but it entrusts the MAC layer with the task of channel access coordination. In particular, PacketOPP exploits the concept of *opportunistic gain* as defined in Section 3.2.1 to estimate the potential throughput improvement associated to each packet transmission, and to decide about future forwarders.

The component that mostly differentiates PacketOPP from equivalent schemes is the internal packet scheduler used to enhance the benefits of opportunistic routing in the presence of multiple simultaneous flows traversing the same node. As illustrated in Figure 13, each node has two transmission queues, one for local traffic (i.e., packets generated by the node itself) and another for overheard packets to be retransmitted (i.e., the forwarding queue). The separation of the input queues for

the two types of traffic is useful for ensuring that sufficient bandwidth is available for the forwarding traffic. Then, PacketOPP employes a packet scheduler to award a higher priority to the packets that are expected to deliver the maximum opportunistic gain, thus maximizing the benefits of opportunistic forwarding also with multiple simultaneous flows. However, for the protocol to be practical, PacketOPP has to address additional design issues. First, it must avoid that starvation may affect some packets that experience lower opportunistic gains that other packets, thus receiving a low priority from the scheduler. Moreover, to reduce the probability that the transmission of the packet with the largest gains is not suppressed by the reception of duplicates, a suitable prioritization of the channel access would be highly desirable. We discuss the solutions adopted in PacketOPP for these issues in following sections.

Packet scheduling and forwarding based on opportunistic gain

As mentioned above, most of the PacketOPP components relies on the concepts of opportunistic gain, initially defined in Section 3.2.1. As pointed out earlier, the opportunistic gain can be interpreted as the potential throughput gain that an opportunistic transmission can provide with respect to a traditional unicast transmission. On the other hand, the opportunistic ratio is used to verify if a new opportunistic transmission positively or negatively affects the throughput improvements accumulated so far.

The quantities defined in equations (3.1) and (3.2) are used in PacketOPP to drive the routing process. Specifically, we need to address the following issues in our design: 1) in which circumstances a node should accept to forward a received packet ; 2) which packet should be forwarded first; and 3) when to transmit the packet. In the following, we describe our approach to cope with these design issues.

Forwarding decision In PacketOPP, the decision rule establishing whether a node is eligible to forward the received packet or not follows the same approach adopted in MaxOPP (3.2.1), thus it depends on

whether condition 3.3 holds. Basically, we impose a minimum throughput gain at each traversed hop by allowing transmissions only from recipients belonging to routes that can provide a gain over the shortest unicast path. Moreover, we require this gain to be at least a percentage $gamma$ of the gain at the previous hop, as defined in equation 3.3, so as to preserve the gain obtained so far. We remind to Section 3.2.1 for a deeper explanation.

Scheduling packet transmissions When a node is chosen as a forwarder for that received packet, the packet is buffered in the forwarding queue, otherwise it is discarded. The opportunistic scheduler defines the policy used to serve the stored packets. It is important to observe that the goal of this work is not to propose an innovative queuing packet serving discipline, which is a topic extensively investigated over more than two decades (PG93). In this paper we aim at demonstrating that combining opportunistic routing with a low-complexity packet scheduler can lead to a more efficient support of multiple simultaneous flows. Therefore, we assume that a simple Round Robin (RR) discipline is used to serve packets stored in the local and forwarding queues. However, to award a higher priority to the packets that may provide a greater throughput improvement we implement the forwarding queue as a *priority queue*, where the packet priority is its opportunistic gain, as defined in expression (3.1). In this way, when the scheduler serves the forwarding queue, it always picks the packet with the highest opportunistic gain. It is worth noting that the RR discipline is the simplest approach to avoid starvation of forwarded traffic due to local saturated traffic. However, in principle it is still possible that a flow characterized by high opportunistic gains will dominate all the other flows going through a node. We can observe that a packet with a large opportunistic gain is usually received over unreliable links. Thus, a node is expected to receive much less packets with large opportunistic gains than packet with low opportunistic gains, and this should mitigate the risk of flow starvation. However, starvation can be also avoided by introducing some randomness in the packet insertion in the priority queue. For instance, using a RED-like (FJ93) approach a

newly received packet can be inserted in the forwarding queue as the last packet with probability p, or according to its priority with probability $1-p$. In this case, p is a system parameter used to control the fraction of packet managed according to a non-FIFO discipline.

In most opportunistic protocols, such as in (RSMQ09; YYW$^+$05), overhearing is generally used as a low-overhead mechanism to suppress the transmission of duplicate copies of the same packet. More precisely, each packet generated by the source carries an incremental sequence number, and each node stores the sequence numbers of its recently forwarded packets. In this way, when receiving a packet the node can determine whether it is a duplicate or not, and it discards the packet accordingly. PacketOPP further extends the overhearing technique because it checks if it has the same packet in its forwarding queue. In this case, if the stored packet is characterized by lower opportunistic gain than the newly received packet it removes it, otherwise the received packet is dropped. In this way, PacketOPP ensures that the packet with a better opportunistic gain get forwarded.

Prioritization of channel access Overhearing is a simple technique to coordinate forwarding nodes. However, it is most efficient when the packet with the highest opportunistic gain is the first to be transmitted, because it can suppress the largest number of duplicate copies, as well as maximize the benefit of the opportunistic transmission. Motivated by these observations, PacketOPP introduces a prioritization of the channel access trying to give a stochastic precedence in the channel access to transmissions with high opportunistic gains. To achieve this goal, PacketOPP slightly modify the backoff procedure at the MAC layer. More precisely, the 802.11 access method randomizes the transmission of broadcast frames by picking up a backoff timer in the range $[1, CW_{broadcast}]$. Since the 802.11 technology does not implement collision detection for broadcast frames, the contention window is constant. To simply support a finite number of priority levels for broadcast transmissions we can use a linear quantization by splitting the contention window into a set of fixed intervals obtained by dividing the maximum window size by the

number of supported priority levels. In other words, if r is the desired number of priority levels, $\Delta_r = CW_{broadcast}/r$ is the duration of these time intervals. Then, we can associate a priority level l (l is a discrete value between 1 and r) to a broadcast transmission by selecting a random backoff timer in the range $[1, l \times \Delta_r]$. This ensures higher chances for channel access to the packet with a lower priority value. Finally, to assign the priority level to a packet we consider its opportunistic ratio. More precisely, let us consider a packet sent by node j and received by node i with an opportunistic ratio $OR_{SD}(j, i, k)$. Since, the higher the opportunistic ratio and the lower should be the priority level l, for simplicity of notation in the following we consider the inverse of the opportunistic ratio. As described in Section 3.2.1, a node discards a received packet if the opportunistic ratio is lower than γ. This implies that the inverse of the opportunistic ratio for any buffered packet is upper bounded by $1/\gamma$. Then, we can quantize the inverse of the opportunistic ratio by dividing it in r fixed interval of size $\omega_r = \frac{1}{\gamma \times r}$. Finally, each packet stored in the forwarding queue receives a priority level l is equal to:

$$l = \left\lceil \frac{1}{OR_{SD}(j, i, k)} \cdot \frac{1}{\omega_r} \right\rceil . \tag{3.4}$$

Loss recovery As explained in section 3.2, in opportunistic routing protocols it is important to employ loss recovery mechanisms also at the routing layer. In PacketOPP we adopt the same approach followed in MaxOPP and (RSMQ06). Specifically, the destination node sends to the source nodes periodic end-to-end unicast acknowledgement messages over the shortest path[2], which contain a fixed size bitmap with the sequence numbers of missing packets. After receiving an acknowledgment packet, the source node retransmits the lost packet. However, to keep the retransmission overhead limited, in PacketOPP a packet can be retransmitted at most once. Consequently, PacketOPP can only provide best-effort reliability to some extent. We remind to section 3.2 for more details.

[2]In our simulations the acknowledgement period is 100 ms.

3.3.2 Performance Evaluation

In this section, we evaluate the performance of PacketOPP using ns-2 (The09) simulations in a set of representative network scenarios. Furthermore, we compare it against two other non-scheduled opportunistic routing protocols, ROMER (YYW+05) and MaxOPP (3.2), and a traditional link-state routing protocol, in our case OLSR (CJ03), using ETX (DCABM03) as link-cost metric.

Simulation set-up

To compare the various routing protocols and collect performance measurements, we have used ns-2 (version 2.33) (The09), a popular event driven simulator that implements the full protocol stack of a MANET. To be able to conduct experiments in controlled environments, while reproducing the behaviors of lossy wireless links, we extended the physical channel model of ns-2 to generate packet losses by dropping packet received at the MAC layer with a constant probability. Note that in our test we use DCF scheme of 802.11 by fixing the transmission rate at 11 Mbps, and disabling the RTS/CTS access method, which is the default setting in most wireless networks.

Concerning the traffic patterns, we model traffic flows as CBR UDP connections generating fixed-size packets at a rate sufficiently high to saturate the wireless channel. If not otherwise stated, the packet size is 1000 bytes. Each simulation run consists of 300-second data transfer, but throughput measurements are not collected during the first 150 seconds to let the ETX metrics converge. In order to collect statistics (i.e., average values and 95% confidence intervals) we replicate each simulation ten times.

Simulation results

In the following simulations we set the γ value to 0.8 in both MaxOPP and PacketOPP to ensure similar forwarding node decisions. Regarding ROMER, we have used the same setting adopted in (YYW+05), which suggests to set the forwarding probability at each node equal to 0.2.

Moreover, our simulations will consider a range of network scenarios, starting from a basic chain topology with a single flow to more complex grid topologies with multiple simultaneous flows.

Single Flow To investigate the effectiveness of PacketOPP in exploiting weak links that may exist between a transmitter and distant nodes, we consider a linear chain topology with a single flow. In this topology, each 1-hop link is ideal, i.e., with perfect delivery rate, and packet can get lost only due to collisions. On the other hand, 2-hop links have a constant delivery rate p_d, which is varied from 0 to 1. For clarity, Figure 7 shows an illustrative 4-hop chain topology. To quantify the performance of each routing algorithm, we used the average end-to-to end goodput, defined as the average rate of non-duplicated packets received at the destination. Figure 14 shows the performance of each routing protocol for a flow that traverses a 7-hop chain topology versus the packet delivery rate p_d. Interesting observations can be derived from the shown results. First of all, OLSR is the worst routing protocol under moderate packet loss rates. This can be explained by noting that the Hello-based link monitoring mechanism used in OLSR to estimate the link quality, provides oscillating measures, especially for intermediate values of the delivery rate (i.e., $0.4 < p_d < 0.8$). As a result, OLSR suffers from excessive route flapping, which causes noticeable performance degradations. These behaviours have been also observed in real network deployments, as reported in (RSBA07). On the contrary, the simulation results indicate that ROMER is more robust than classical shortest path routing protocol against link quality variations, because ROMER delivers redundant data copies in a randomized manner over the candidate forwarding mesh, ensuring better resiliency against lossy links. However, we can observe that ROMER performance slightly depends on the delivery rate of 2-hop links. In addition, ROMER performs noticeable worse than OLSR for delivery rates close to either 0 or 1. This can be explained by observing that in the current design ROMER employs a fixed forwarding probability. Therefore, the total number of forwarded copies is constant and depends only on the number of transmitters' neighbors. However,

for smaller values of the delivery rate, most of these packet copies get lost before reaching the destination, because there are a few transmissions that reach 2-hop neighbors. In other words, the static forwarding probability does not ensure that a sufficient number of packets are successfully received by the destination when the channel conditions do not offer enough transmission opportunities. On the other hand, for larger values of p_d, the overhead due to the delivery of redundant copies is excessive because path diversity is limited and multiple transmissions cause throughput penalty over traditional unicast routing. Moreover, it is worth reminding that ROMER uses a credit-based scheme to further limit the number of transmissions. However, setting a static credit budget may not be efficient to cope with different network conditions. The results shown in Figure 14 confirm that MaxOPP's throughput is significantly higher than OLSR and ROMER in all the considered scenarios, and the maximum throughput gain exceeds 600% for OLSR and 250% for ROMER. This is mainly due to the ability of MaxOPP to adaptively select the most advantageous forwarder at each hop. As expected, in the single flow scenario PacketOPP has similar performance than MaxOPP because forwarding node selection is based on the same principles in both algorithms, and the opportunistic packet scheduler cannot fully exploit the diversity of transmission opportunities, which mainly arises with multiple flows. Nevertheless, PacketOPP slightly increases the flow throughput over MaxOPP, especially for small values of the delivery rate p_d, because, as discussed in Section 3.3.1, the access prioritization employed by PacketOPP protocol improves the efficiency of the overhearing technique.

Figure 15 compares the average throughput achieved by PacketOPP, MaxOPP, ROMER and ETX-based OLSR for linear chain topologies by varying the number of wireless hops between the source and the destination nodes and considering intermediate delivery rates for 2-hop links (i.e., $p_d = 0.5$). Interestingly, the graph shows that ROMER performs better for shorter chains (i.e, $2/3$ hops) than for longer chains (i.e., $5/6$ hops). This can be explained by noting that in a short chain there are a very few opportunities to take advantage of weak links, and most of the

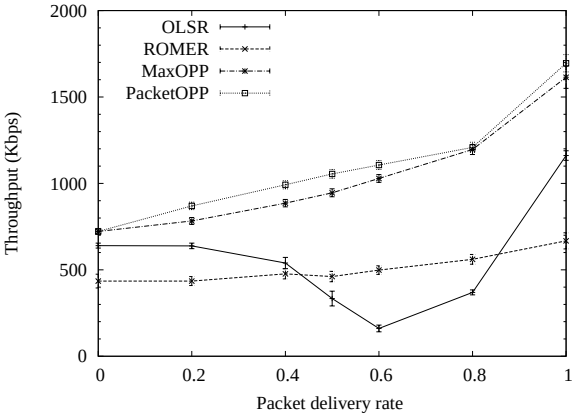


Figure 14: Flow throughput in a 7-hop chain topology as a function of the packet delivery rate p_d.

opportunistic gain comes from the use of broadcast frames, which have no ACK overhead, while the shortest path routing uses unicast transmissions and incurs ACK overhead. In these conditions, there are no substantial differences among the various opportunistic algorithms, which achieve similar performance. Nevertheless, the greater efficiency of MaxOPP and PacketOPP algorithms over ROMER becomes rapidly evident as soon as we consider longer chain topologies.

Multiple Flows In this section, we investigate the performance of multiple flows in grid topologies. Specifically, we consider a 5×5 grid-based network topology where 1-hop links are perfect (i.e., delivery rate equal to one), while 2-hop links have intermediate delivery rates (i.e., $p_d = 0.5$) as shown in Figure 10. Regarding the traffic scenarios, we randomly choose source and destination pairs, however a node can be either the source or the destination of a single flow. For each scenario, we report the total mean throughput and the mean fairness index averaged over ten different traffic patterns. The system fairness is measured using the classical Jain's index (Jai84), defined as $(\sum x_i)^2/(n \cdot \sum x_i^2)$, where x_i is the goodput of the i-th flow and n is the number of flows.

Figure 16(a) and Figure 16(b) show the total average throughput and

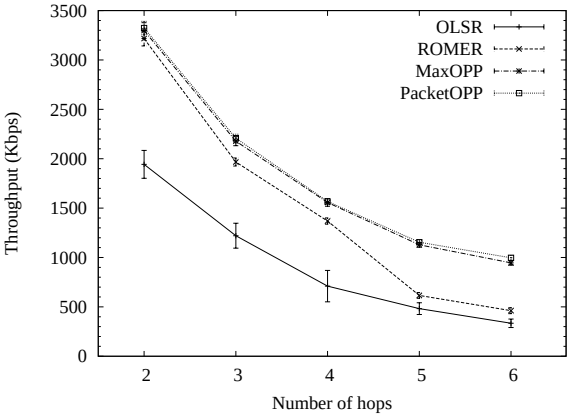
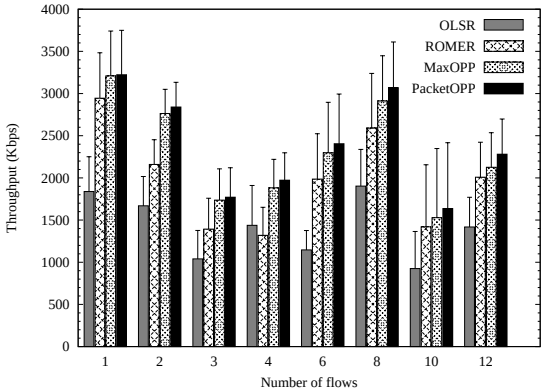
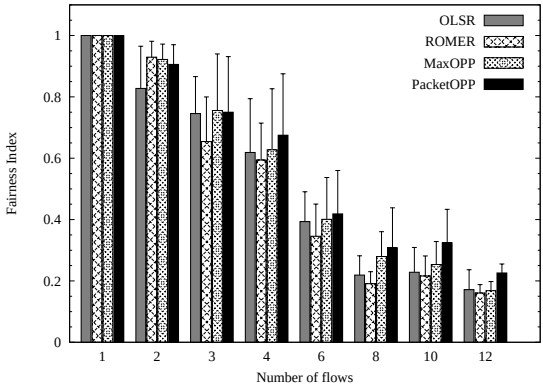


Figure 15: Throughput in a chain topology versus the number of hops for $p_d = 0.5$.

the fairness, respectively, for different number of flows. First of all, we can observe that the confidence intervals are quite high. This is due to the fact that, although we fix the number of flows for each scenario, the structure of each traffic pattern used in the simulation runs is highly diverse because we pick randomly the source and destination nodes. Furthermore, the results confirm that, as the number of flows increases, the system fairness degrades because flows traversing a greater number of hops are disadvantaged with respect to shorter flows. The graphs also clearly indicate that opportunistic routing strategies provide significant throughput gains over traditional unicast routing mainly because opportunistic routing algorithms can cope well with unreliable wireless links, and are more responsive to variable channel conditions. However, as shown in Figure 16(a), PacketOPP outperforms both MaxOPP and ROMER for any number of flows, with the maximum throughput gain varying from 70% for OLSR, 50% for ROMER and 10% for MaxOPP depending on the traffic scenario and the congestion level. Regarding the fairness properties, the percentage improvements ensured by PacketOPP over the other routing protocols are even higher, especially when there are many simultaneous flows. These results demonstrate the ability of



(a) Throughput



(b) Fairness

Figure 16: Total average throughput (a) and fairness (b) in a 5×5 grid topology versus the number of flows.

PacketOPP to select the forwarders that may provide the maximum potential throughput gain, and to effectively schedule the packet with the highest opportunistic gain in such a way that they have a higher chance to access the wireless medium first.

Chapter 4

Using Localized Context to Improve Reliability and Adaptability of Opportunistic Routing

4.1 Combining end-to-end with localized data

In the previous chapters, we have presented and discussed several opportunistic routing approaches, and highlighted their benefits and limitations in mitigating the negative impact of channel quality variability on WMNs performance. Moreover, we have proposed two novel algorithms able to select at run-time the paths that maximize the throughput gain.

Although the opportunistic routing approach has shown its potentiality to deal with high loss rates, the inherent variability of channel quality over space and time requires a high degree of flexibility and adaptability to changing network conditions when building the best path. When opportunistic routing is used, routes are constructed in a hop-by-hop fashion, evaluating the potential packet progress provided by each recipient. Most of the existing opportunistic routing solutions

base the forwarder selection on end-to-end principles (BM05; WLL08). More specifically, packets are opportunistically forwarded through multiple alternative paths that are deemed more convenient than the shortest unicast path. Typically, the cost from each potential forwarder that can be used to reach the destination is compared with the cost of the shortest path that would be used by traditional unicast routing, and only the nodes ensuring a performance improvement over the shortest path are selected as potential forwarders. Although end-to-end characteristics are fundamental to rank all the available multiple paths for each source-destination pair, local network conditions are necessary to perform more accurate and efficient forwarding decisions at each hop. Indeed, the cost of a path is not uniformly distributed over space, nor constant over time, hence even two equal-cost paths might present significantly different link quality distributions one from the other (CRSK06). This disparity might be considerable especially in challenging scenarios, where some sub-regions of the network may be more disadvantaged than others. Hence, a localized decision (i.e. hop-by-hop) is not sufficient to perform the most appropriate routing decision when only end-to-end principles are considered.

Motivated by the above considerations, in this chapter we propose RELADO (RELiable ADaptive Opportunistic routing protocol), an innovative routing paradigm able to combine end-to-end with local information, so as to adapt the forwarding decisions to the localized context observed at the time of packet reception. In a nutshell, RELADO relies on an end-to-end opportunistic routing paradigm to select the best forwarder(s) at run-time and on a per-packet basis. In addition, real-time local channel information permit to adapt per-packet routing decisions to short-term and localized (i.e., space-limited) channel dynamics. Thanks to the combination of end-to-end and localized information, RELADO is able to promptly react to the channel quality variations along the whole path, mitigating the negative effect of sub-optimal decisions due to the lack of a complete and up-to-date path view.

4.2 RELADO design

In this section we firstly present an overview of the main principles behind RELADO design, and then we describe the algorithm in details.

4.2.1 Overview

The primary goal of RELADO is to ensure robust communications even in case of transient failures, heterogeneous link quality distributions or challenged channel conditions, by dynamically adapting routing decisions to both end-to-end and localized network context. In common to other opportunistic routing schemes, each transmitting node (i.e. source node and forwarder) broadcasts received packets, and any neighboring node overhearing this transmission is a potential next hop. The fundamental issue in opportunistic routing is how to select the next hop(s) for each transmitted packet and how to coordinate multiple forwarders without incurring high overhead and packet duplicates. Our aim is also to provide each forwarder with the additional capability to estimate the criticality of each transmission, that is, to evaluate the importance of its role as a forwarder in building a reliable path for each specific packet. To achieve the above goals with a distributed low-overhead algorithm, we need to address two main issues: 1) each receiving node must be able to autonomously evaluate its potential contribution to the forwarding process, in order to decide whether to transmit the packet or not, and 2) the contribution must take into account two components: efficiency and resilience. The former represents how the forwarder can contribute to reduce the cost to reach the destination, thus increasing the throughput, considering the path from an end-to-end perspective. The latter represents how the transmission of this forwarder is crucial to ensure that the packet makes progress towards the destination, even in case of lossy links. This component is needed to adjust forwarder selection based on the localized channel qualities and network state, and their impact on packet transmissions. This approach attempts to provide the same level of robustness at each hop of the traversed path, regardless of the channel

conditions. In the following, we explain how RELADO achieves those goals, and then we provide a more detailed description of the algorithm. RELADO does not use any pre-determined list of forwarders, but it relies on a distributed algorithm that provides each node with the capability to autonomously decide whether to participate to the forwarding process. To achieve this goal, as we proposed for the MaxOPP forwarding procedure (3.2.1), each node receiving a packet evaluates the potential throughput gain that it would generate by further broadcasting the received packet. More specifically, each node evaluates its effectiveness as a forwarder for the received packet by comparing its expected cost towards the destination with the remaining cost along the shortest path. This step is required to ensure that packets traverse only paths with a cost lower (or at most equal) than the shortest path, thus providing a throughput gain. On the other hand, resilience to critical and variable channel conditions can be guaranteed only with a flexible and adaptive routing approach, which must be able to easily detect critical but localized channel conditions. Thus, the basic routing scheme described above should be enhanced to provide a certain degree of flexibility depending on the channel qualities observed in the range of each transmission.

To better explain this concept, let us consider the example in Figure 17, where node F is forwarding a packet destined for D. In this context we can identify three categories of neighbors for node F: 1) *admissible* forwarders, 2) *resilient* forwarders, and 3) *excluded* forwarders. The first set includes all node F's neighbors with a difference between their cost towards the destination and the corresponding cost along the shortest path lower or equal than a given threshold δ. This implies that the admissible forwarder can participate to the packet forwarding. On the contrary, any excluded forwarder will drop the packet because its cost to the destination is higher than that from F, meaning that the packet will get farther from the destination. An important role is played by the second set, which is formed by the remaining node F's neighbors. The importance of resilient forwarders resides in the possibility they offer to enlarge or restrict the set of admissible forwarders according to the criticality of each transmission. More precisely, if the probability that a

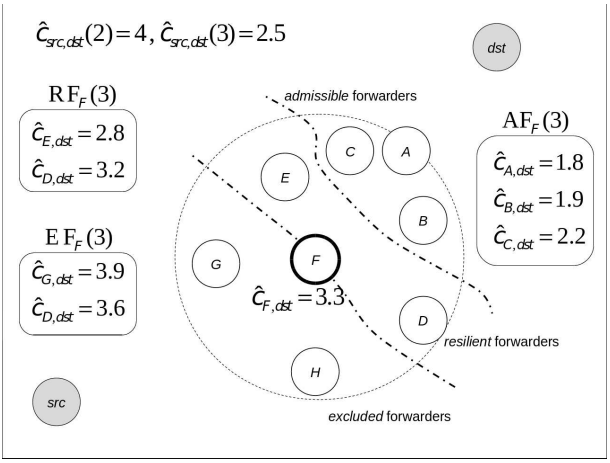


Figure 17: Example illustrating how the neighbors of node F can be divided into the sets of admissible, resilient and excluded forwarders. In this example we assume that: *i)* node F has received a packet that has already traversed 2 hops (thus k=3), *ii)* node F acts as a forwarder for the received packet, which is rebroadcasted to all node Fs neighbors, and *iii)* $\delta = 0$.

packet transmitted by F reaches one of the admissible forwarders is sufficiently high (i.e., above a given threshold), then the current choice of the threshold δ is appropriate. On the other hand, if the localized channel conditions are critical, we need to guarantee that additional resilient forwarders will be included in the admissible set. The resulting augmented set permits to increase the reception probability from node F to the nodes closer to the destination. Although this flexibility might reduce the throughput gain obtained so far, in such challenging situations we clearly need to trade path efficiency for resilience. In the following sections we explain in details how to compute all the quantities needed to determine the various sets of forwarders, and to estimate the criticality of a packet transmission.

Opportunistic routing based solely on end-to-end principles

As stated above, RELADO inherits the basic end-to-end routing approach that we have proposed in 3.2.1 for the MaxOPP protocol. Thus, in this section we briefly overview its main principles to better clarify the differences with the new scheme. We highlight that the goal of this work is to develop and evaluate an adaptive opportunistic routing protocol and compare it with another opportunistic routing scheme lacking of the capability to adapt to local conditions, in order to evaluate the effectiveness of the additional feature. Thus, our aim is not to select the best opportunistic approach available, but to select an efficient opportunistic protocol and to provide it with the reliability an adaptivity necessary to improve its efficiency in most of the typical wireless mesh network scenarios. For this reason, we select MaxOPP as the basic end-to-end opportunistic scheme and we extend it with additional capabilities, hence we compare RELADO with MaxOPP.

As explained in section 3.2.1, if the opportunistic gain(3.1) is positive and sufficiently high, it means that the packet has reached a node unexpectedly closer to the destination than its intended next hop, because it has traversed a path much shorter than the minimum-cost path. Indeed, it is convenient to take advantage of this favorable condition by allowing the node i to play the role of the k-th forwarder. Moreover, if the packet has traversed long hops, significantly improving the path throughput compared to the shortest path, then successive transmissions should not nullify the benefit obtained so far. Thus, the routing protocol should admit only those forwarders that can preserve this gain. On the other hand, we should also allow enough forwarders to be activated, in order to guarantee packet progress. This second goal is achieved by specifying the minimum value of the opportunistic ratio (3.2, 3.3). This ratio represents the gain variability from the (k-*1*)-th to the k-th forwarder along the path. In MaxOPP, whenever a node receives a packet, it checks two conditions: *i)* the opportunistic gain must be positive; *ii)* its opportunistic ratio must be higher than the gain threshold, which is decided by the source node. In MaxOPP the opportunistic gain takes into account only

end-to-end features (i.e., the characteristics of the paths traversed by the received packets.), while in this study we advocated the idea that better reliability (and throughput) can be achieved if the localized context is considered in a more explicit way. Hence, in RELADO design, we still use the concept of opportunistic gain, but we opt for a less restrictive and more flexible approach. In the next section, we describe in details how this is achieved in RELADO.

Opportunistic routing combining end-to-end principles with localized context

As explained in the RELADOs overview, when a node i rebroadcasts a packet that has already been transmitted (*k-1*) times, all node i's neighbors can be divided in three different groups. There is the set of admissible forwarders $AF_i(k)$, which is defined as the set of node i's neighbors that provide a gain over the shortest unicast path greater or equal than δ_i. Formally, $AF_i(k)$ can be obtained as

$$AF_i(k) = \{v \in N(i) \mid cost(p_{S,D}; k) - cost(p_{v,D}) \geq \delta_i\} \tag{4.1}$$

where *N(i)* is the set of *i*'s neighbors. Note that the set of admissible forwarders with $\delta_i = 0$ corresponds to the set of eligible forwarders used in MaxOPP. The set of excluded forwarders $EF_i(k)$ contains node i's neighbors with a cost towards the destination higher than that from node i itself, and is given by

$$EF_i(k) = \{v \in N(i) \mid cost(p_{v,D}) - cost(p_{i,D}) > 0\} \tag{4.2}$$

Finally, the set of resilient forwarders $RF_i(k)$ consists of the remaining neighbors of node *i*, that is

$$\begin{aligned} RF_i(k) = &\{v \in N(i) \mid cost(p_{v,D}) \leq min(cost(p_{i,D}), \\ &max(0, cost(p_{S,D}; k) - \delta_i))\} \end{aligned} \tag{4.3}$$

Since we have assumed that each node has a global knowledge of the network, which is provided by the underlying link-state dissemination protocol, node *i* can easily compute the above three sets. However, if we assume that the constant δ_i is piggybacked in the header of each packet transmitted by node *i*, then each node *i*'s neighbor that correctly receives a packet sent by node *i* can also easily determine to which set it belongs to. Now, the fundamental open issue is how to set the δ_i threshold to obtain the desired level of routing reliability. To this end we introduce the concept of *transmission robustness*. More specifically, the robustness of a packet transmission carried out buy node *i*, is defined as the probability that the transmitted packet will be correctly received by at least one of the nodes in the set $AF_i(k)$ of admissible forwarders (which in turn depends on the δ_i threshold). It is interesting to note that this value represents also the probability that the transmitted packet can proceed towards its destination, since only the admissible forwarders are allowed to transmit the packets they receive. To compute the transmission robustness, let us denote with $p_{i,j}$ the probability that a packet transmitted by node *i* is correctly received at node *j*. Clearly, $(1 - p_{i,j})$ is the probability that the packet is lost on the link $e = (i, j)$. Then, the probability $\theta_i(k)$ that a packet transmitted by node *i*, which was received at node *i* after having traversed (*k-1*) hops, reaches at least one of the nodes in the set $AF_i(k)$ can be computed as follows

$$\theta_i(k) = 1 - \prod_{j \in AF_i(k)} (1 - p_{i,j}) \tag{4.4}$$

By definition the $\theta_i(k)$ value is the robustness of transmissions performed by node *i*, and this metric can be used to quantify the criticality of a packet transmission, and to decide if additional nodes should be employed as forwarders to help the packet progress. To clarify this point, let us consider again the example in Figure 17. Considering the path costs reported in the figure, it is immediate to obtain from formulas 4.1, 4.2 and 4.3 with $\delta_i = 0$ that $AF_F(3) = \{A, B, C\}$, $RF_F(3) = \{D, E\}$ and $EF_F(3) = \{G, H\}$. Let us suppose that, with this choice of admissible forwarders, the transmission robustness $\theta_F(3)$ is 0.5, which means that a

packet should be transmitted on average two times to ensure that it is received by at least one node in $AF_F(3)$. For many applications such value of reliability can be too low. On the other hand, allowing other nodes (i.e., D and E) to forward the packet can improve routing resilience. From a practical point of view, let $\omega_i(k)$ be the minimum transmission robustness we want to guarantee at node i for each packet that has already traversed (*k-1*) hops in the network. Algorithm 1 shows the pseudo-code of the algorithm that is used in RELADO to compute the $AF_i(k)$ set and the threshold δ_i that guarantees (whenever feasible) $\theta_i(k) \geq \omega_i(k)$.

Algorithm 1 Algorithm performed by a node *i* to compute the $AF_i(k)$ set and the threshold δ_i that guarantees $\theta_i(k) \geq \omega_i(k)$

Input: $N(i), k, \omega_i(k), D, S$
Output: $AF_i(k), RF_i(k), EF_i(k), \delta_i$

1: $\delta_i \leftarrow \infty$
2: $\Omega = \emptyset$
3: $\Pi = \{j \in N(i) \mid cost(p_{j,D}) \leq cost(p_{i,D})\}$
4: $\theta_i(k) \leftarrow 0$
5: **while** $(\theta_i(k) < \omega_i(k)) or (\Pi \neq \emptyset)$ **do**
6: $\quad v \leftarrow \min_{v \in \Pi}\{cost(p_{v,D})\}$
7: $\quad \Pi : \Pi \backslash \{v\}; \Omega = \Omega \bigcup \{v\}$
8: $\quad \theta_i(k) \leftarrow ComputeRobustness(i, \Omega)$
9: $\quad \delta_i = cost(p_{S,D}; k) - cost(p_{v,D})$
10: **end while**
11: $AF_i(k) = \Omega$
12: $RF_i(k) = \Pi$
13: $EF_i(k) = N(i) - \Omega - \Pi$

The algorithm starts setting to infinity the minimum opportunistic gain needed to activate a forwarder (line 1). At this point, the set Ω including the admissible forwarders is empty (line 2), while the set Π listing the resilient forwarders includes all node *i*s neighbors that are closer than node i itself to the destination (line 3). Then, we extract from set Π the resilient forwarder v with the smallest remaining cost to reach the destination and we add it to the set Ω (line 7). Then, we re-compute the transmission robustness, as defined in equation 4.4, with the enlarged set

Ω (line 8), and we update the threshold δ_i to ensure that node v will be considered as an admissible forwarder. We continue to process the nodes in the set Π until it is not empty or the transmission robustness fulfils the constraint on the desired level of routing resilience, i.e., $\theta_i(k) \geq \omega_i(k)$. With Algorithm 1, we are able to provide a flexible routing solution that adaptively trades throughput maximization for communication robustness.

Loss recovery

As stated in the previous chapters, one of the fundamental issues in opportunistic routing is the design of an efficient and low-overhead loss recovery mechanism. For these reasons, most of the existing opportunistic routing solutions introduce special link-layer retransmission mechanisms, or rely on end-to-end retransmissions implemented at upper layers (network or transport). The former permits to recover from losses while packets proceed from source to destination, whereas the latter informs the source about packets that have not been received within a reasonable time period and must be retransmitted. However, end-to-end acknowledgement messages impose a delay on packet recovery, and hop-by-hop retransmissions might introduce a considerable overhead. In order to provide both forms of reliability, we use a combination of the above two mechanisms by limiting the disadvantages derived from each approach. Thus, similarly to MaxOPP and PacketOPP, we introduce periodic end-to-end ACK messages with a fixed-size bit map reporting the sequence numbers of not received packets, and additional information on flow performance used at the source node to dynamically set protocol parameters (such as gain ratio threshold γ). The use of a fixed-size bit map permits to keep ACK-related overheads limited. In addition, we provide a controlled level of data redundancy along the path through the simultaneous activation of multiple forwarders, which act as a distributed hop-by-hop loss recovery mechanism generating lower overhead than traditional link-layer retransmissions. As explained above, the

number of forwarders is tuned according to the reliability of the channel during the actual transmission.

4.2.2 Performance Evaluation

In this section, we evaluate the throughput performance of RELADO and compare it with two alternative protocols. For the purpose of the evaluation, we implement it in the ns-2 network simulator, and we conduct an extensive set of simulations.

Simulation setup

We consider a WMN of 25 nodes randomly placed within an area of 800m x 800m. Each node is equipped with one omnidirectional antenna. Concerning the MAC layer, we used the 802.11 DCF scheme and we fix the transmission rate to 11 Mbps. Moreover, we disabled the RTS/CTS access method, which is the common setting in most wireless networks. With regard to the physical layer, instead of the classical TwoRay propagation model, we use the Shadowing model, which is more suitable to represent realistic outdoor environments. More precisely, the Shadowing model expresses the received power at a wireless interface as a random variable that takes into account the effect of multi-path propagation. In particular, we set the shadowing deviation to 4 and the path loss exponent to 2, in order to simulate an outdoor environment (ABB$^+$04). If not otherwise stated, the receiving threshold of the network interface is set so that 95% of the packets are correctly received at the distance of 100m. Finally, the data traffic is modeled as constant bit-rate UDP flows between randomly selected node pairs. More details about traffic patterns are provided in following sections.

In order to evaluate the effectiveness of our routing approach, we compare it with OLSR (CJ03) and MaxOPP 3.2. The former scheme is used as a representative of link-state single-path routing protocols, whereas MaxOPP is an example of opportunistic protocol that employs a randomized forwarding scheme to select the next hop(s) at run-time and on a per-packet basis.

Simulation results

In this section we firstly discuss the results obtained in single-flow scenarios with static topology and stable channel conditions. Then, we show the performance obtained in more critical scenarios due to node failures or abrupt degradation of channel qualities. Finally, we analyze multi-flow scenarios.

Single-flow scenario with stable conditions Firstly, we evaluate RELADO performance when only a single flow is active in the network, in order to gain a clearer insight on the effectiveness of our approach on flows with different characteristics (number of traversed hops, channel quality of traversed links, etc.). In order to ensure flow diversity, we study the performance of the three protocols with 50 different source-destination pairs. Each source node sends 1000-byte payload packets at 1Mbps data rate for 1000 seconds. Finally, we execute 10 independent runs for each simulation to be able to compute average values and confidence intervals of the throughput of all flows. Note that simulation warm-up lasts 200 seconds, and statistics are collected in the remaining 800 seconds.

Figure 18 shows the distribution of flow throughputs for the three considered protocols. We use a box plot with whiskers, since it permits to easily display the most significant quartiles of the distribution: the band inside the box is the median, the bottom and top boundaries of the box are the first and the third quartile, respectively, and the ends of the whiskers are the minimum and the maximum values. The square inside the box represents the average throughput. As shown by the throughput distributions, our approach offers clear improvements over the other two algorithms. First of all, the average throughput obtained with RELADO is higher than with the other two protocols, as MaxOPP is able to outperforms OLSR of about 33%, whereas RELADO offers an average throughput improvement over OLSR around 46%. Furthermore, OLSR shows the lowest minimum throughput, and the largest range between minimum and maximum values. By inspecting the simulation traces we

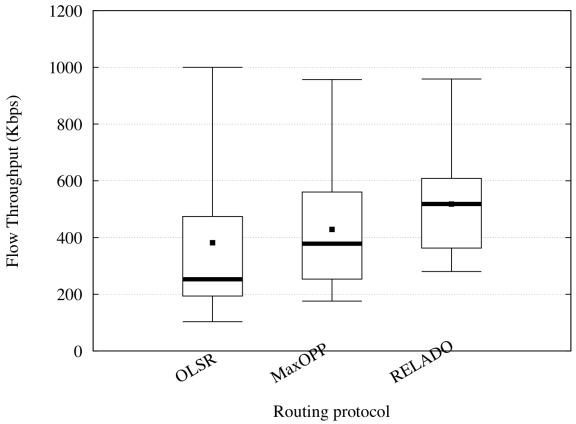


Figure 18: Boxplots of throughput of 50 node pairs in a 25-node random network for different routing schemes. Squares represent the mean throughputs.

discovered that OLSR usually provides the highest throughput when short-path flows are considered (one or two hops from source to destination). However, it greatly suffers from an increase in the number of traversed hops, which causes severe performance degradation. It is important to note that a high variability of the throughput obtained by individual node pairs is unavoidable due to the significant differences in the length of shortest routes connecting such pairs, which span from single hop traditional routes to routes that have six hops. However, Figure 18 indicates that with OLSR many flows have low throughputs, which causes a quite low median throughput. On the other hand, the opportunistic routing paradigm is able to exploit more transmission opportunities than OLSR, especially for long paths, improving the throughput of disadvantaged flows. In particular, the median throughput obtained with RELADO is significantly higher than both OLSR and MaxOPP. This means that, with our scheme, a considerable percentage of flows (50%) has a significantly higher throughput than with conventional algorithms. Finally, let us consider the statistical dispersion of throughput values, i.e., the range between the first quartile and the third quartile, also known as interquartile range. We can observe that, when RELADO is used as the routing protocol, the throughput is concentrated over significantly higher values, and even the throughput obtained with the least favorable conditions (minimum) is noticeably higher.

Single-flow scenario with node failure In order to evaluate the performance of our algorithm in critical scenarios, we firstly consider the case in which a node unexpectedly disconnects from the network, thus all the other nodes are required to update link-state information and re-compute routing tables. In order to create a particularly challenging environment, for each source-destination pair we disconnect a random node located on the shortest path connecting source and destination nodes, so that its failure certainly affects the performance of the activated flow, requiring the computation of a new sequence of nodes to reach the final destination. Note that the wireless mesh backbone is basically a static network, and mesh routers are usually not resource-limited as de-

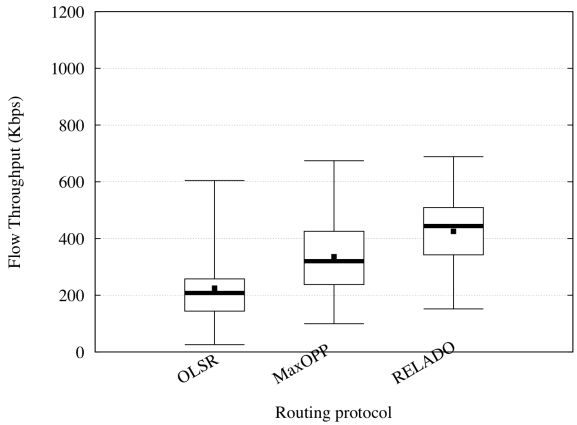


Figure 19: Impact of node failures on the throughput of 50 node pairs in a 25-node random network for different routing schemes.

vices in many other wireless networks (such as sensor networks). However, route disruption is possible even in such static environments, given the challenging conditions under which a WMN can be deployed, as discussed in Section 4.1. Hence, our intent is to analyze how the considered protocols react to such topology changes. Figure 19 reports the distribution of flow throughputs for the three considered protocols, measured after the node failure.

The most evident result shown in Figure 19 is the severe performance degradation experienced by all protocols when even one node stops to relay packets (compared to the values reported in Figure 18). In particular, OLSR shows a very low minimum throughput value, as well as the lowest maximum throughput value, which confirms the negative effect of topology changes on traditional link-state routing schemes. Moreover, with OLSR the interquartile range is quite small and shifted towards low throughput values, whereas opportunistic routing algorithms show significantly better performance. The main reason of this noticeable per-

formance degradation is the delay due to the computation of an alternative path. In addition, the new path will generally consist of longer links affected by higher loss rates. On the other hand, MaxOPP and RELADO are able to quickly adapt to the changed network topology, and to take advantage of any transmission opportunity to deliver packets around the failed node. Indeed, even during the collection and dissemination of updated link-state information, opportunistic routing can continue packet forwarding based on the data already available, since it already uses multiple alternative paths. For all these reasons, OLSR suffers from route disruption more than MaxOPP and RELADO. Finally, it is important to note that RELADO significantly outperforms not only OLSR but also MaxOPP, providing an interquartile range more shifted towards higher throughput values. This performance gains can be explained by considering the fact that RELADO reacts to the localized worsening of channel and connectivity conditions by increasing the set of admissible forwarders, thus enhancing its resilience against higher packet losses.

Single-flow scenario with channel quality degradation In this section, we show and discuss the impact of link quality degradation on the performance of OLSR, MaxOPP and RELADO. More specifically, at the beginning of the simulation the receiver threshold of the network interface is set so that 95% of the packets are correctly received at the distance of 100m. Then, after 100 seconds the receiver threshold is suddenly decreased to 0.80, meaning that only 80% of packets are correctly received at the distance of 100m. The quality decreases similarly in all links, causing a more uniform degradation of network conditions with respect to a node failure, which causes a more localized perturbation of network state. Thus, this event might cause the re-computation of many minimum-cost paths, and it certainly generates a uniform increase of link loss rates. Figure 20 reports the distribution of flow throughputs for the three considered protocols, measured after the change of link qualities. As shown in the figure, OLSR is characterized by the largest difference between the flow with the highest throughput and the most disadvantaged flow achieving the minimum throughput. The reason is that link-

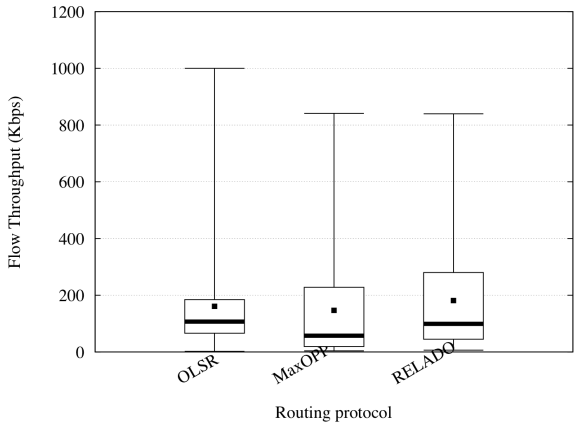


Figure 20: Impact of channel quality degradation on the throughput of 50 node pairs in a 25-node random network for different routing schemes.

layer retransmissions allows OLSR to provide a more efficient loss recovery for flows with one or two hops, as opposed to opportunistic routing that does not find enough alternative paths to face the drop in channel qualities. However, as path length increases, OLSR performance noticeably decreases, whereas opportunistic routing schemes find more opportunities to transmit. Hence, although the OLSR first quartile and median throughput are slightly higher than with MaxOPP and RELADO, the latter protocols outperform the traditional single-path link-state routing in terms of the interquartile range. In particular, RELADO shows a significantly higher average throughput and third quartile than both OLSR and MaxOPP. We remind that the third quartile is an important metric because it represents the throughput trend for most of the flows (i.e., 75% of collected samples). Again, this can be explained by observing that RELADO reacts to the worsening of link qualities by accordingly increasing the set of potential forwarders to guarantee stable transmission robustness.

In conclusion, compared to the other routing schemes, RELADO is clearly able to boost throughput to higher values in both static and dynamic situations. OLSR suffers from route disruption more than opportunistic routing, because path re-computation is a costly and long process. On the other hand, a homogeneous channel quality variation seems to be a challenging case for all the considered routing protocols, leading to severe performance degradation. However RELADO is the most effective scheme in properly reacting to these network changes, limiting their negative effects.

Multiple-flow scenario In this section we consider more scaled scenarios, where a certain number of flows is simultaneously activated, to increase the offered load in the network. As in the single-flow case, we randomly select source and destination for each activated flow and we start all flows at the same time, in order to observe how these concurrent transmissions affect the performance of the three routing protocols. Note that flow patterns are chosen in such a way that each node can be at most the source or the destination of one flow. Figure 21 reports average values and 95% confidence intervals for the network capacity (i.e., the sum of the throughputs of individual flows) versus the number of flows for the three considered routing protocols. These results are obtained repeating each test with twenty different combinations of selected flows.

The plot indicates that RELADO significantly outperforms OLSR in all the considered scenarios, with throughput gains ranging from 25% to 55%. On the other hand, RELADO provides good improvements with respect to MaxOPP for low numbers of flows (up to 20%), and attains comparable performance for higher numbers of flows. It is interesting to note that the network capacity is maximal when there are five flows in the network, and further increasing the number of flows causes a decrease of the network capacity. This can be explained by noting that by adding more flows in the network there is a tradeoff between the improvement in link utilization and the increased inter-flow interference. The variability between the cases with eight and ten flows can be explained by considering that, in order to simulate realistic scenarios, data

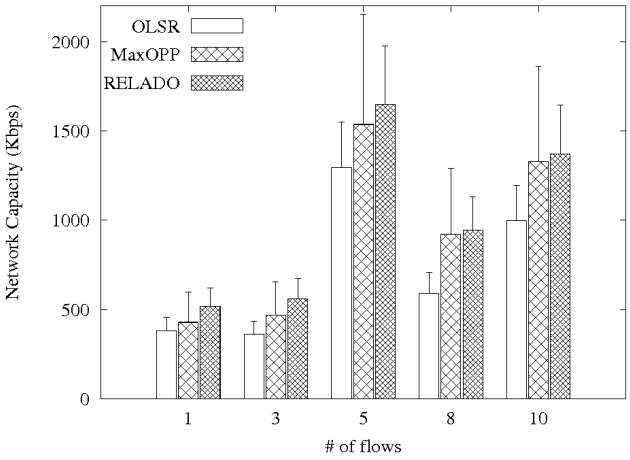


Figure 21: Impact of the number of simultaneously active flows on the total network capacity.

flows are selected randomly so as to ensure a high variability of path characteristics. Hence, the scenario with eight flows is not guaranteed to reach the network capacity as in the ten flows case. Indeed, our purpose is to evaluate the performance of the proposed protocol under different and realistic conditions.

Chapter 5

Machine Learning-based Hybrid Routing for WMNs

5.1 Self-adaptive selection of the best routing strategy

As discussed in the previous chapters, opportunistic routing has been envisaged as a promising routing paradigm to mitigate performance degradation due to the unpredictability and unreliability of wireless transmissions. Although the encouraging results obtained so far, the community still lacks a comprehensive study of the conditions under which opportunistic outperforms traditional routing. In this chapter, we are not concerned with the analysis of which are the scenarios where one specific routing solution for WMNs shows superior or worse performance than another specific routing solution. On the contrary, the central question addressed in this chapter is: *How to design self-adaptive mesh networking solutions that autonomously and on-the-fly decide upon the best routing protocol for a traffic flow from a set of supported algorithms, given an estimate of the current network state?* In principle an exact answer to this question could be obtained through a comprehensive routing model characterizing the impact of routing primitives, traffic patterns, network topologies and link characteristics on the network performance. As a

matter of fact, theoretical analysis of the capacity of WMNs has received much attention in recent years (e.g., (ZWR08; ZLZ08; ZJ09)). However, these studies are either limited to asymptotic bounds or based on centralized and computationally complex models. Furthermore, existing results cannot be applied to heterogeneous environments, where each node may use a different routing strategy.

To answer the above questions, in this chapter we propose a novel routing architecture that exploits *reinforcement learning* to allow each node to autonomously choose the best forwarding strategy from a predefined set of routing protocols as network conditions (e.g., traffic loads or link qualities) change. The idea of using machine learning techniques to solve specific routing problems in ad hoc networks is not new (see (For07) for a survey). However, to the best of our knowledge there is very little research in applying reinforcement learning for the optimal composition of different routing strategies. The rationale behind the use of reinforcement learning is that it enables the design of decision agents that learn through *trial-and-error* interactions with a dynamic environment how to behave in order to maximize the reward associated to their actions (SB98). For the purpose of evaluation, in this chapter we design an intelligent agent able to combine a traditional unicast routing protocol with an opportunistic routing scheme. A recent paper (BGLAO09) has investigated how to combine on-demand routing and delay-tolerant routing[1] in intermittently-connected ad hoc networks. However, the solution proposed in (BGLAO09) is basically an extension of the AODV routing protocol (PBRD03) to cope with network partitions. On the contrary, the approach proposed here is more general because it allows the combination of multiple and arbitrary routing algorithms without making any assumption on the characteristics of the underlying network.

In the following, we first formulate the network state space for our decision problem with the objective of identifying the minimum number of state variables required to estimate the routing efficiency. Second, we

[1]Delay-tolerant routing is a different type of opportunistic routing, where transmission opportunities are generated by node mobility rather than wireless diversity.

define a set of rules to instruct the learning agent how to behave in order to maximize its reward. It is important to note that the reward function is used to formalize the goal of the learning problem. For instance, the reward function can give higher value to policies that improve throughput performance, reduce delays or increase reliability. This results in a highly configurable system able to customize the routing behaviour according to different application objectives and QoS requirements. Finally, the tool we utilize to solve the learning problem is the *Q-learning algorithm*, which is a dynamic programming method that works by continuously improving its estimates of the value of particular actions at particular states (WD92). The key properties of Q-learning are: $i)$ it is a *model-free method*, i.e., it does not require any model of the system, $ii)$ it converges with probability one to the optimal policy under general assumptions, and $iii)$ it is very easy to implement using lookup tables. On the negative side, it may converge slowly to the optimal policy depending on the state size. Thus, to improve efficiency and speed up convergence we rely on compact network-state representations and smart initialization of action-value functions.

5.2 Background on Reinforcement Learning

Reinforcement Learning (RL) is a popular machine learning technique, which allows an agent to automatically determine the optimal behaviour to achieve a specific goal based on the positive or negative feedbacks it receives from the environment after taking an action (SB98). More formally, let us assume that the interactions between the agent and the environment occur at a sequence of discrete time instants t. Following the same notation as in (SB98), the learning problem can be formulated by defining:

- The *state* $s_t \in S$ of the environment as observed by the agent, where S is the set of possible states.
- The *action* $a_t \in A(t)$ chosen by the agent, where $A(t)$ is the set of actions admissible at time t.

- The probabilistic *reward* $r_{t+1} \in \mathcal{R}$, whose mean value is provided by the reward function $R(s_t, a_t)$. Roughly speaking, the reward function maps the state-action pair to a scalar value, and it is used to measure the goodness of taking action a_t in state s_t.

- The *state transition* function $T(s_t, a_t, s_{t+1})$, which provides the probability of making a transition from state s_t to state s_{t+1} after performing action a_t. Note that $T(s_t, a_t, s_{t+1})$ captures the non-determinism of the environment, because taking the same action on the same state may result in different next states.

Typically, in machine learning it is assumed that the state transition probabilities satisfy the Markov property, i.e., they are independent of any state or action previous to time t. In this case, the environment where the agent operates is described through a Markov Decision Process (*MDP*), and several optimized learning algorithms have been studied for this class of environments[2]. More specifically, let us denote with π a *policy*, which is a mapping between states and actions. In other words, the policy defines the probability $\pi(s, a)$ that the learning agent takes action a when in state s. In some cases, a policy can be a simple deterministic function, but arbitrarily sophisticated policies are possible. In general, the main goal of the learning agent is to find in the set of feasible policies the optimal one providing the maximum long-term reward. To express this in a mathematical setting we define the *expected return* W_t as a function of the sequence of rewards received after time step t. In the case of learning tasks that are continuos a typical formulation for the return functions is as follows

$$W_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \ , \tag{5.1}$$

where γ is a parameter ($0 \leq \gamma < 1$) called *discount rate*, used to determine the present value of future rewards. It is intuitive to observe that if $\gamma = 0$,

[2]It is important to note that RL can also deal with non-Markovian environments (SB98). However, Markov property is a good approximation for many network characteristics observed over long time intervals.

the agent will behave so as to maximize its immediate reward, even if this would imply a lower long-term return. It is also important to note that other models exist for defining the return function (e.g., the finite horizon model, in which the agent should optimize its expected reward over a fixed number of time steps), which may be more suitable for other learning problems, e.g., cyclical tasks.

Now, the solution of any RL problem can be formulated mathematically in an MDP perspective and under the discounted infinite horizon optimality model described in (5.1) by introducing the concept of *optimal value* $V^*(s)$ for each state $s \in S$. More precisely, $V^*(s)$ is defined as the expected return if the agent starts at state s and then executes the optimal stationary policy π^*. From the previous definition it follows that

$$V^*(s) = \max_{\pi} E\left(\sum_{t=0}^{\infty} \gamma^t r_t\right) . \tag{5.2}$$

In the MDP case, it is a consolidated theoretical result that the optimal value function (5.2) is unique and it can be computed as the solution of the following set of equations (SB98):

$$V^*(s) = \max_{a}\left(R(s,a) + \gamma \sum_{s' \in S} T(s,a,s')V^*(s')\right) . \tag{5.3}$$

Equation (5.3) provides the analytical basis for developing simple iterative methods for calculating $V^*(s)$ and π^*, assuming that $R(s,a)$ and $T(s,a,s')$ are known. However, the main idea behind RL is that it is possible to obtain the optimal policy for an MDP environment even when the model of the environment is not known or difficult to learn. Thus, several *model-free* RL methods have been developed, which are based on the concept of *action-value functions* (SB98). More formally, the optimal action-value function $Q^*(s,a)$ is the expected return if the agent starts at state s, executes action a and follows the optimal policy π^* hereafter. Owing to this definition, it holds that $V^*(s) = \max_a \ Q^*(s,a)$. Hence, $Q^*(s,a)$ can be derived recursively as follows

$$Q^*(s,a) = R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') \max_{a'} Q^*(s',a') . \tag{5.4}$$

It is useful to note that from the knowledge of the $Q^*(s,a)$ function, the learning agent can straightforwardly build the optimal policy by simply choosing the action with the highest value at each state. More formally, this means that $\pi^*(s) = argmax_a Q^*(s,a)$.

A fundamental development in the context of model-free RL methods is the *Q-learning algorithm* (WD92), which updates the action-value function using the following rule

$$Q(s,a) = Q(s,a) + \alpha \left[r + \gamma \max_{a'} Q(s',a') - Q(s,a) \right] , \qquad (5.5)$$

where s' and r are the state and reward, respectively, observed after performing action a in state s, and α is a positive step-size parameter determining the *learning rate*. It has been shown that under general assumptions[3] the Q-learning algorithm will asymptotically converge with probability one to the optimal action-value function Q^* independently of the agent's behaviour, i.e., of the policy π being followed (WD92).

The great advantage of Q-learning algorithm is that it is typically easier to implement than other RL techniques. On the negative side, it may converge slowly to the optimal policy due to the so-called *exploration* and *exploitation* issue. Basically, when in state s the learning agent should exploit its accumulated knowledge of the best policy to obtain high rewards, but it must also explore actions that it has not selected before to find out a better strategy. Moreover, each action must be tried many times to gain a reliable estimate of its expected reward. In the literature there are a variety of popular exploration heuristics, ranging from the simple *greedy* strategy, which selects action a in state s that maximizes the current estimation of $Q(s,a)$, to more sophisticated stochastic techniques, which assign a probabilistic value for each action a in state s according to the current estimation of $Q(s,a)$. In Section 5.3.2 we further discuss suitable action selection strategies for the application scenario addressed in this work.

[3]The most important condition is that all state-action pairs are visited infinitely often.

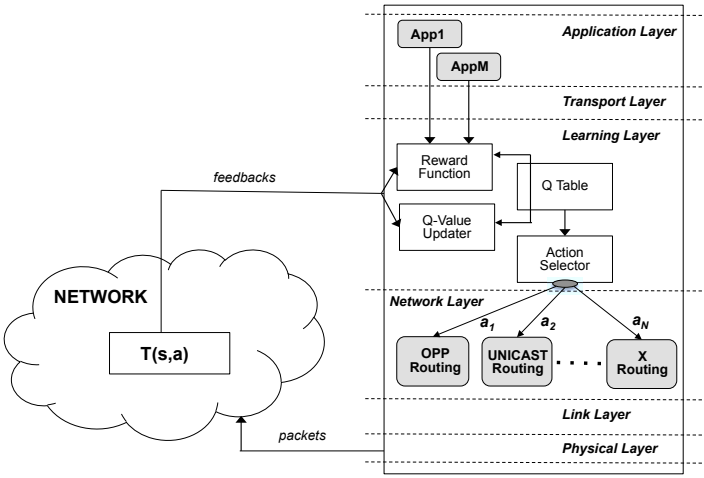


Figure 22: Node architecture

5.3 Routing Protocol Design

In this section we first provide a high-level overview of the various components in our routing framework, and then we describe how to apply reinforcement learning techniques for improving routing efficiency in WMNs.

5.3.1 Routing architecture

Figure 22 shows the main components in the node architecture. A key assumption in our routing framework is that each mesh node implements a number of pre-defined routing algorithms, and it can decide on-the-fly which protocol to activate. The decision on which routing strategy should be used can be made at different levels of granularity. For instance, the source node may decide the best routing algorithm on a flow-level basis. A finer granularity in the optimization process would be obtained by using a combination of routing algorithms for each connection, for instance dividing the packet flow into batches and choosing the best routing algorithm on a batch-level basis.

The core component in the architecture depicted in Figure 22 is the reinforcement learning agent implementing the Q-learning algorithm. Such agent operates between the transport layer and the network layer. Specifically, the agent receives *feedback signals* from the network, which are used both to estimate the network state, and to compute the (positive or negative) reward generated by the last action taken by the agent. Such feedback signals can be in the form of observed network quantities, or explicit messages sent by the destination and intermediate nodes. It is important to note that the reward function formalizes the goal of the agent and it is application dependent. Indeed, the application scenario sets the QoS requirements for the route, which are mapped in a suitable reward function. For instance, in case of applications generating bulk data transfers, throughput is the most important performance metric and the reward should be a function of the measured connection throughput. In case of delay-sensitive traffic alternative formulations for the reward would be needed. From the knowledge of network states and transition rewards, the agent can update the Q-value for the last transition using formula (5.5). Then, the Q-values are represented by a two-dimensional lookup table indexed by state-action pairs. Note that lookup tables are very easy to implement, although the memory requirements may be prohibitive in case of huge state spaces.

The last, but very important, learning component is the action selector. As explained more in detail in the following, in our model an action is a change in the fraction of packets in a flow that should be forwarded using one of the available routing protocols. The simplest solution would be to always select the routing protocol that has provided the highest throughput up to the time of the agent decision. However, there is a trade-off between short-term gains and long-term rewards. Furthermore, the agent has to execute each state-action pair infinitely often in order to guarantee the convergence to the optimal Q^* matrix. In the following sections, suitable exploitation strategies will be defined.

5.3.2 Network state, reward and actions

In the rest of this section, to simplify the protocol presentation, we assume that each mesh node implements two different routing strategies, namely, an unicast routing protocol and an opportunistic routing protocol. Note that our solution is not restricted to any specific routing algorithm and it could be easily extended to more than two routing options. Furthermore, we assume that the agent on the source node operates on batches of packets all generated by the same traffic flow rather than on single packets in order to reduce the cost for network state monitoring.

In our model, an *action taken by the source is one possible choice for the* η_i *parameter*, defined as the fraction of packets in batch i to be sent using the unicast routing protocol (obviously, the remaining $(1-\eta_i)$ packets are sent using the opportunistic routing protocol). In other words, the learning agent at the source node dynamically adapts the portion of traffic sent using unicast routing or opportunistic routing depending on its view of the network state and the received network feedbacks. More precisely, the destination collects throughput measurements for the packet received from each batch[4]. Then, let ρ_i^U be the throughput of packets from batch i delivered using the unicast routing protocol, while let ρ_i^O be the throughput of packets from batch i delivered using the opportunistic routing protocol. Thus, $\rho_i = \rho_i^U + \rho_i^O$ is the total throughput for packets of batch i measured at the destination. In addition, let η_i' be the fraction of the overall throughput at the destination for batch i due to the unicast transmissions. More formally, this can be expressed as follows

$$\eta_i' = \frac{\rho_i^U}{\rho_i} \ . \tag{5.6}$$

Now, we can introduce the concept of *routing efficiency* ϵ_i, a 3-valued vari-

[4]We assume that a tiny header is added to the traditional routing header containing information such as the batch id and the type of routing algorithm used to forward that packet.

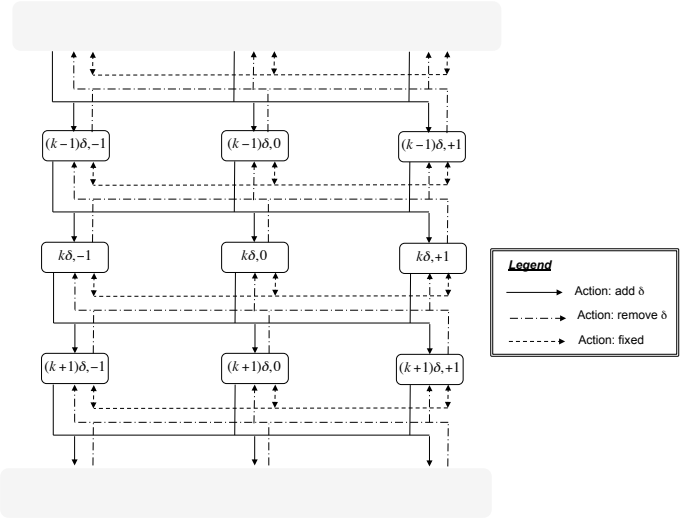


Figure 23: Portion of the transition graph used in our learning-based routing agent – note that each arrow (s, a, s') should be labelled with the reward value $R(s, a)$ and the transition probability $T(s, a, s')$. However, for the sake of figure readability these values are not reported in the diagram.

able defined as follows

$$\epsilon_i = \begin{cases} 1 & \eta_i' > \eta_i \\ 0 & \eta_i' = \eta_i \\ -1 & \text{otherwise} \end{cases} . \tag{5.7}$$

In other words, the ϵ_i quantity measures if the packets delivered using the unicast routing protocol suffer from less, the same number, or more losses than packets of the same batch delivered using the opportunistic routing protocol. Now we can define the *network state s_i as the two-dimensional tuple* $\{\eta_i, \epsilon_i\}$. Note that our objective is not to obtain global network information, but to define an approximate compact representation of the network state to infer the impact of agent's actions on the performance of each traffic flow.

The last element of our reinforcement learning system is the *reward*

$R(s_i, a_i)$ *gained when in state* s_i *the agent performs action* a_i. Intuitively, for bulk data transfers the immediate reward should depend on the total throughput obtained at the end of the transmitted batch of packets. More precisely, let us assume that the source receives at the end of the transmission of batch i a feedback message with the ρ_i^U and ρ_i^O measures[5]. Then, the agent chooses a new value for the proportion η_{i+1} between unicast and opportunistic transmissions in the next batch $(i+1)$ (see Section 5.3.3 for a detailed discussion on the action selection policies). At the end of the $(i+1)$-th batch, the agent will receive a new feedback message and it will calculate the reward as follows:

$$R(s_i, a_i) = \rho_{i+1} + \omega(\rho_{i+1} - \rho_i) \ , \tag{5.8}$$

where $0 < \omega < 1$, is a constant parameter. As expressed in formula (5.8), the reward consists of two contributions. The first term is intuitive: the higher the measured throughput, the higher is the reward gained by the source node. However, the agent should also assign higher value to actions that provide an increase in the flow throughput, and penalize actions that decrease the throughput. This means that the reward should also depend on the throughput obtained in the previous step. To this end, we introduce the second term $\omega(\rho_{i+1} - \rho_i)$, which is a scaled measure of routing gain[6]. Note that definition (5.8) does not violate the MDP assumption.

Whenever the source node receives a feedback message from the destination or a neighbor node with the information needed to compute the reward associated to its last action, it updates its estimates of the action values using formula (5.5). It is important to note that for learning problems in stationary environments it is usual to set the learning rate α as a decreasing function of time. The reason is that at the beginning of the agent's life the rate α should be large enough to quickly overcome initial conditions. As the estimates of the optimal action values converge,

[5]For the sake of simplicity we assume that the feedback signal is not delayed. In Section 5.3.4 we discuss on the impact of delayed feedbacks, which is the normal case in real-world networks.

[6]In our simulations $\omega = 0.5$

decreasing the rate α mitigate unnecessary fluctuations. However, it is intuitive to note that the routing problem considered in this work is intrinsically *non-stationary* because the network conditions can change over time (e.g., due to interference, node failures, etc.). In such cases it is useful to give more weight to recent rewards than past ones to guarantee a fast adaptation to dynamic network behaviours. This is achieved by using a *constant* learning rate (in our implementation $\alpha = 0.5$, which provides a good tradeoff between reactivity and stability). For the same reason, we also set the discount rate γ equal to zero, to give more importance to recent rewards than to future ones, which may be associated to a network with changed conditions.

Before concluding this section, it is important to explain how our learning agent reduces the number of admissible state-action pairs to explore. Indeed, the time the Q-learning agent needs to converge to the optimal policy will be affected by the size of the network state S and the number of feasible actions. First of all, we restrict the parameter η to be a real number in the range $[0, 1]$ obtained as a multiple of a fixed positive constant. More formally, let $(K+1)$ denote the maximum number of positive and equally distributed values that η can take in the range $[0, 1]$. By definition it follows that:

$$\eta = k\delta \quad \text{with} \quad k = 0, 1, 2, \ldots, K \tag{5.9}$$

where $\delta = 1/K$. Furthermore, for each η, we can have three different conditions for the efficiency parameter (i.e., $-1, 0, 1$). This implies that the overall size of the state space S_t is $3 \times (K+1)$. In our implementation we have set $K = 10$, which means that the fraction of packets in a batch that is transmitted using unicast routing changes with a step size equal to 10 percent. We have investigated the impact of the parameter K on the network performance and we found that our proposed protocol performs equally well with larger K values. In other words, the network performance are negligibly affected by the use of a finer granularity in the allocation of packets to the different routing options.

As described in Section 5.3.1, in our model an action is the selection of a new η value for the successive batch of packets. In principle, after each

packet batch, the learning agent could select any of the admissible values of η in the range $[0, 1]$. However, such approach generates a number of state-action pairs that is quadratic with K, which may lead to unacceptable convergence delays. Consequently, to reduce the number of possible state-action pairs we assume that in state $\{\eta_i, \epsilon_i\}$ the learning agent can only increase/decrease the η_i parameter by δ, or maintain it constant. In other words, the agent's actions can only cause transitions to adjacent states. Based on these requirements Figure 23 depicts the transition diagram from a generic state $\{k\delta, \epsilon_i\}$. As shown in the figure, at most nine transitions are possible from each state. This means that the total number of state-action pairs per each flow destination that we should rate with a Q-value is bounded by $27 \times (K + 1)$. Note that this number is small enough to make acceptable the use of lookup tables for implementing the Q-learning algorithm.

5.3.3 Action selection policy

A very important learning procedure is the action selection policy. In the following simulations, three intuitive and quite popular policies will be tested. It is important to note that there is not an action selection strategy that is a priori better then others, but it depends on the properties of the task to be accomplished.

- $greedy$: The greedy method is the simplest action selection rule because, at time t it always chooses the *greedy* action a^*, namely the action with the maximum estimated action-value function. More formally this can be expressed as follows:

$$Q(s_t, a^*) = \max_a Q(s_t, a) \tag{5.10}$$

  where s_t is the network state at time t. This method is guaranteed to maximize the immediate reward but it ignores exploration. In other words, a pure greedy action selection policy never samples actions with non-maximum Q value, although they could lead to action selections with better long-term return.

- $\varepsilon - greedy$: A simple alternative to the pure greedy strategy is a *near*-greedy selection methods. More precisely, the learning agent will behave most of the time greedily, selecting an action with maximum expected value with probability $(1-\varepsilon)$, but with a small probability ε instead selects an action uniformly over the set A_t, and independently of the action-value estimates. The advantage of an ε-greedy method is that it guarantees that every action will be sampled an infinite number of times as the time goes to infinity, which is the necessary condition for the convergence of the Q-learning algorithm. However, the advantage of ε-greedy over greedy methods highly depends on the network behaviour. For instance, if the variance of reward samples is very low, exploration may not be necessary to find the optimal action. Indeed, the greedy method would know the almost exact value of each action after trying it once. In this case the greedy method might actually perform better than an ε-greedy method because it would quickly find the optimal action and then never explore.

- $softmax$: One drawback of the ε-greedy method is that it has the same probability of choosing worst-case actions as well as best-case actions when in exploration phase. An obvious solution for this issue is to use softmax action selection rules, which rank and weight actions according to their value. The most common softmax function used in reinforcement learning to convert values in action probabilities is the following (SB98):

$$\pi(s_t, a_t) = \frac{exp[Q(s_t, a)]}{\sum\limits_{a' \in A_t} exp[Q(s_t, a')]} \tag{5.11}$$

  Based on (5.11), there can be a great difference between the selection probabilities of different actions depending on their values.

5.3.4 Practical issues

In the previous sections, we have described the general design of our routing framework. But for the protocol to be practical, there are additional challenges, which we discuss in detail below.

Delayed feedbacks

The destination is responsible for collecting the throughput measurements and sending them to the source. Since most of the opportunistic routing protocols use end-to-end acknowledgements from destination nodes for signalling purposes, we can assume that those messages are extended to also deliver the statistics needed for the source's learning agent. However, in this case the feedback signal will be asynchronous with respect to the sequence of batches of packets generated by the source. Alternatively a new dedicated control message could be generated at the end of the batch transmission. In any case, the source's agent cannot receive the feedback signal (due to end-to-end transmission delays) in time for selecting the η value for the successive batch. In addition, sending a feedback message after each received batch would generate an excessive overhead. To address this issue, in our implementation the destination generates a feedback message only after receiving m consecutive batches of packets. Furthermore, this allows the agent to apply averaging to the per-batch throughput measurements, thus mitigating fluctuations due to transient conditions. Then, the source uses the same η value for at least $m+1$ consecutive batches. Our results indicate that $m=10$ provides a good tradeoff between responsiveness, stability of throughput estimates and efficiency.

Packet losses

Packets can get lost in the network for several reasons. Generally, unicast routing protocols are totally unaware of these losses that are recovered through mechanisms implemented at different layers of the protocol stack (e.g., using layer-2 or layer-4 retransmissions). On the contrary, most of the opportunistic routing protocols directly retransmit lost packets. Since the η parameter is a measure of the routing efficiency for individual packet transmissions, retransmitted packet should not be included in such computation.

Implementation issues

There are several details to take into account during practical implementation. First of all, we must decide a suitable size for the batch. Note that, differently from other routing schemes that operate on batches of packets, such as ExOR (BM05), it is not necessary that the source collects a full batch of packets before starting the forwarding process. In our implementation, the batch is a virtual concept used to facilitate the learning process, and to easily allocate each packet to one of the two routing options so as to respect the η fraction. Thus, each packet is forwarded as soon as it reaches the head of the transmission buffer. In our implementation, the batch size is set equal to $(K+1)$ packets.

A second concern regards the measurement of the routing efficiency. In real-world networks the instantaneous throughput may be fluctuating due to a variety of causes (e.g., transient changes of link quality due to fading variations, burstiness of packet arrivals, randomness of channel accesses, etc.). Consequently, it may be difficult to obtain stable estimates of $\epsilon = 0$, as defined in (5.7). For these reasons, in our implementation we adopt an extended definition of routing efficiency as follows:

$$\epsilon_i = \begin{cases} 1 & \eta'_i/\eta_i > 1+\beta \\ 0 & |\eta'_i - \eta_i| \leq \beta\, \eta_i \\ -1 & \text{o.w.} \end{cases} . \tag{5.12}$$

In other words, relationship (5.12) expresses that unicast and opportunistic routing have the same efficiency if the relative difference between η' and η values is less or equal than β. In our implementation, $\beta = 0.1$, which is sufficient to absorb small fluctuations of throughput measurements.

Finally, since the Q-learning algorithm is an averaging method that improves its estimates of the average value of action-state pairs as new actions are taken, it is dependent on the initial values of the action-value estimates. The simplest *initialization* approach would be to set all the initial action values to zero. However, the initial action values could also be used as a simple way of controlling the initial direction of exploration

and to speed up convergence. In general, it is advantageous to use optimistic initial values to encourage exploration even if greedy actions are selected most of the time. Consequently, in our implementation we start a 2-second probing phase at the beginning of each new connection to get an initial estimate of the efficiency of unicast routing. More precisely, let us assume that in the initial probing phase the traffic is sent using only unicast routing (i.e., $\eta = 1$). In addition, let $\widetilde{\phi}^U$ be the offered load per unit time measured at the source and $\widetilde{\rho}^U$ the unicast throughput measured at the destination at the end of the probing phase. Then, $\widetilde{\eta} = \widetilde{\rho}^U / \widetilde{\phi}^U$ is a rough estimate of the efficiency of unicast transmissions. Thus, a good way for calculating an initial guess for the values of the action-state function is to assume that states with $\eta > \widetilde{\eta}$ would not contribute to significantly increase the system reward. More formally, this is equivalent to say that $Q(s, a) = \widetilde{\rho}^U$ for all $s = \{\eta, \epsilon\}$ such that $\eta \leq \widetilde{\eta}$, and $Q(s, a) = 0$ otherwise. Then, the Q-learning algorithm will improve the $Q(s, a)$ estimates converging towards the optimal values.

5.4 Performance Evaluation

A comprehensive simulation study has been conducted to compare the performance of the proposed self-adaptive routing protocol, hereafter simply called *Hybrid*, against two alternative routing schemes. We first show the gains in terms of throughput improvements in single flow scenarios. Then, we analyze scenarios with multiple flows and unreliable channels.

5.4.1 Simulation environment

To carry out the following simulation we use the *ns-2* network simulator, which implements the full protocol stack for multi-hop wireless networks. We consider WMNs of 25 nodes, each of which is equipped with one omnidirectional radio antenna. These static nodes are placed randomly in a 500m$\times$500m area. Concerning the physical layer characteristics, to make more realistic simulations we use the *Shadowing* propaga-

tion model instead of the classical TwoRay propagation model, because it permits to describe the received power at a certain distance as a random variable. This is more appropriate for WMN environments, where multi-path propagation due to large reflectors, e.g., buildings, may induce significant fluctuations of instantaneous transmission and sensing ranges. The following results have been obtained by setting the path loss exponent equal to 2 and the shadowing deviation to 4, which are typical values for outdoor environments (ABB+04). If not otherwise stated the receiving threshold for the network interface is set in such a way to ensure a 95% correct reception rate at the distance of 100m. Concerning the MAC layer, we use the 802.11 DCF scheme with a fixed transmission rate equal to 11 Mbps. Moreover, we have disabled the RTS/CTS access method, since this is the default setting in most wireless networks.

As motivated in Section 5.1 , for the purpose of evaluation we use three routing protocols. First, we use OLSR (CJ03) as a representative of proactive link-state routing protocols. Regarding the opportunistic routing option, we use the PacketOPP protocol (3.3) , a lightweight opportunistic routing algorithm able to select at each hop, and at run-time, the candidate forwarders that can maximize the opportunistic throughput gain. Finally, the third protocol is Hybrid, which is a dynamic combination of OLSR and PacketOPP obtained by applying the framework described in Section 5.3 . In the following simulations, to generate the data traffic we use constant bit-rate UDP flows. We do not use TCP-controlled data transfers because traditional TCP would experience unacceptable performance degradations with opportunistic routing due to the frequent out-of-order packet deliveries (BM05) . Such issue could be mitigated by using sophisticated TCP variants, but this is out of the scope of this chapter. Finally, the broad range of network and traffic scenarios used in the simulations will be described in detail in the following sections.

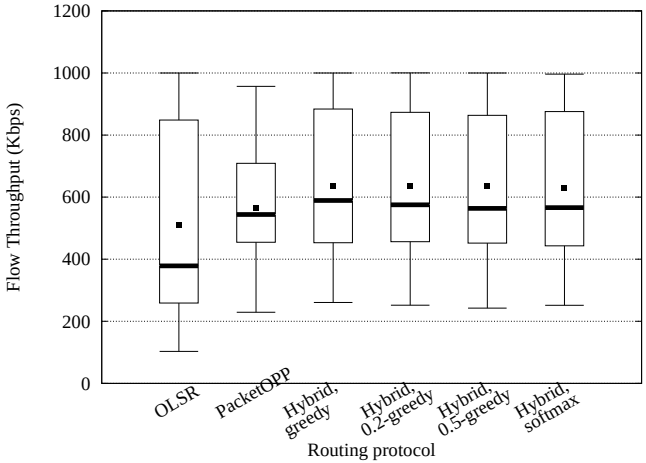


Figure 24: Boxplots of throughput of 80 node pairs in a 25-node random network for different routing schemes. Squares represents the mean throughputs.

5.4.2 Numerical results

Single flow

First, we evaluate the performance of our solution under a single flow scenario. To this end, Figure 24 shows a box-and-whiskers plot[7] depicting the quartiles of the distribution of flow throughput for OLSR, PacketOPP and Hybrid with different action-selection strategies. The results shown in Figure 24 have been obtained as follows. First, we randomly select a node pair in the network as source and destination of an UDP flow, which sends packets with a payload of 1000 bytes at an offered rate

[7]We remind that a boxplot is a way of graphically depicting groups of numerical data through their five-number summaries: the bottom and top of the box are the 25^{th} and 75^{th} percentile, the band in the box is the median (50^{th} percentile) and the ends of the whiskers represent the minimum and maximum of all the data. Note that boxplots are particularly useful to display differences between sets of data without making any assumption on the underlying statistical distribution. Moreover, they provide a more aggregate and concise representation of data variability than classical Cumulative Distribution Functions (CDFs).

of 1 Mbps. Then, ten statistically independent runs are repeated with this node-pair selection to obtain average values and confidence intervals for the flow throughput. Since throughput performance of individual flows are highly variable, we repeat the same set of simulations with 80 different node pairs to collect a number of samples sufficiently large to obtain good estimates of quartile ranges.

Several important considerations can be derived from the above diagram. First, if we consider the min-max interval, traditional unicast routing shows the largest range, and in particular the lowest minimum throughput. By inspecting the traces we discovered that unicast routing achieves very low throughput for source-destination pairs that are separated by many hops. In these cases PacketOPP can take advantage of multiple forwarding nodes and more opportunities to use long links providing throughput gains of a factor of two or more. On the other hand, due to its self-adaptive characteristics, Hybrid protocol is able to select the routing option most suitable for both particularly disadvantaged flows and flows with the highest throughputs. The second observation regards the statistical dispersion of throughput values, i.e., the range between the first quartile and the third quartile, also known as *interquartile range*. It is important to note that a high variability of the throughput obtained by individual node pairs is unavoidable due to the significant differences in the length of shortest routes connecting such pairs, which span from single hop traditional routes to routes that have six hops. However, Figure 24 indicates that OLSR has the largest variability, with many flows having low throughputs. This results in a quite low median throughput. On the other hand, PacketOPP obtains a median throughput 50% higher than OLSR. Despite that, the throughput distribution for PacketOPP is quite concentrated around the median value, which means that only a few flows can achieve high throughputs. On the contrary, Hybrid protocol shows the best performance in terms of median throughput, which is slightly higher than opportunistic median throughput, as well as interquartile range, which is more shifted towards high throughputs. A final observation on the impact of action selection strategies on the Hybrid protocol performance. Figure 24 shows

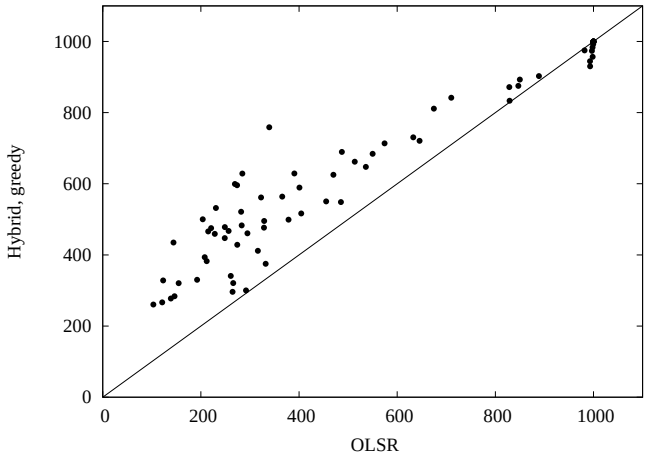


Figure 25: Scatter plot showing the relationship between the throughput of individual flows obtained with OLSR and Hybrid (greedy).

that Hybrid protocol behaves almost equally well with all the considered policies. However, the pure greedy strategy behaves slightly better than the other considered strategies (i.e., ε-greedy methods and softmax method), as it provides the highest median throughput. In the following, all the reported results (except for convergence times) are obtained using the greedy version of our Hybrid scheme.

To better understand the primary reason of throughput gains provided by Hybrid protocol over the other routing strategies, Figure 25 shows the throughput relationship between OLSR and Hybrid with a greedy action-selection policy for the 80 node pairs considered in Figure 24, while Figure 26 shows the relationship between PacketOPP and Hybrid. Figure 25 indicates that there is a large number of node pairs with low unicast throughputs that obtain a significant gain when using Hybrid protocol. Specifically, for half of the node pairs Hybrid outperforms OLSR by more than 50%. Moreover, the node pairs with the highest throughput (mainly pairs with single hop routes) are not affected by the use of Hybrid routing. This means that the overhead introduced by Hybrid (i.e., feedback messages and additional information in packet headers) have a negligible impact. On the contrary, Figure 26 indicates

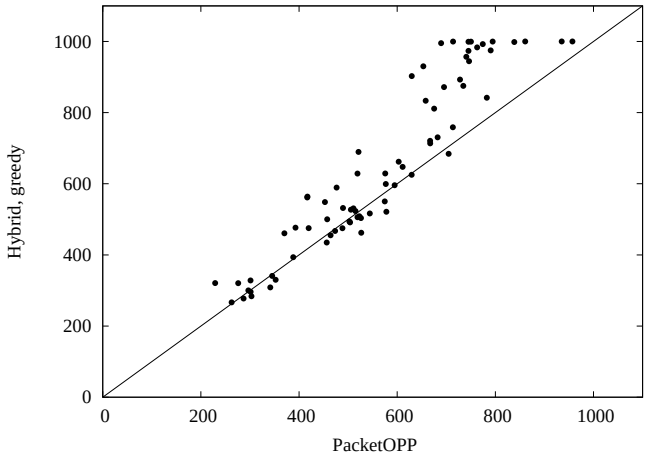


Figure 26: Scatter plot showing the relationship between the throughput of individual flows obtained with PacketOPP and Hybrid (greedy).

that the throughput gain of Hybrid over PacketOPP occurs mainly for node pairs with mid to high throughput values, while it provides similar performance for node pairs with lower throughputs. To clarify the routing selection behaviour, Figure 27 shows the probability mass function (PMF) of the average η value for the 80 considered node pairs and a greedy action-selection policy. To plot such curve we first compute the average fraction of traffic that was delivered using unicast routing for each individual node pair, and then we derive the distribution of such measurement over all the flows. Interestingly we can observe that the PMF of the η parameter shows two principal modes, i.e., two distinct peaks (local maxima). More precisely, most of the flows converge towards η values in the range $[0.25, 0.4]$, which means that these flows use a combination of unicast and opportunistic transmissions, but give preferences to opportunistic routing. Then, there is a second smaller group of flows that converge towards η values close to one, which are mostly node pairs that are separated by a few hops. Interestingly, there is also a non-null probability that a flow uses only opportunistic routing, i.e., $\eta \leq 0.1$.

To conclude this section we present results related to the convergence

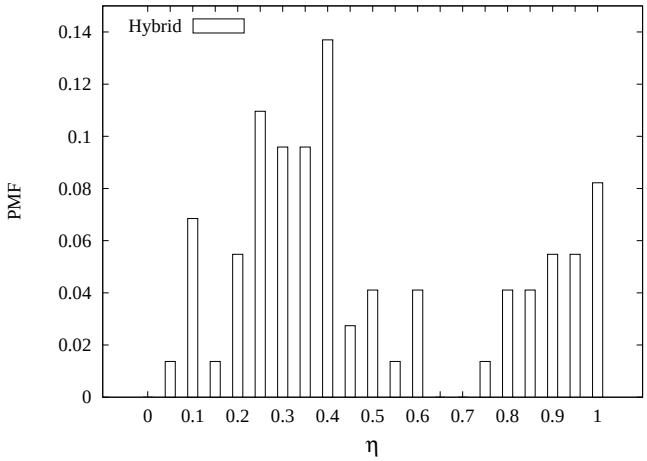


Figure 27: Probability mass function of the average η value per node pair.

Table 2: Comparison of T_1 and σ_1 for different action selection strategies.

	greedy	0.2-greedy	0.5-greedy	softmax
T_1	27.23	33.19	34.87	30.55
σ_1	0.007	0.012	0.016	0.02

time of our Q-learning algorithm. To this end, we define two metrics. The first metric is time T_1, which is defined as the *first hitting time* for the optimal η value. Basically, T_1 measures the time needed for the intelligent agent to discover for the first time the η value providing the best throughput for each individual flow. The second parameter is σ_1, which is defined as the standard deviation of the η values chosen by the agent after T_1. In other words, σ_1 measures the variability of the η parameter around its optimal value. Note that, due to exploration the agent can not persistently use the optimal η value, but it must occasionally test alternative η values. However, a sufficiently small σ_1 value indicates that the agent does not drift significantly from the optimal operating conditions. Table 2 reports T_1 and σ_1 values for different action selection strategies. The results listed in the table indicate that the greedy action selection strategy outperforms the other schemes both in terms of convergence times and stability. In particular, a greedy policy ensures that the η value

providing the maximum throughput is selected most of the time.

Impact of the number of flows

We now turn to consider more scaled scenarios. Basically, we repeat the same simulations discussed in the previous section, but increasing the number of parallel data transfers active in the network. Figure 28 reports average values and 95% confidence intervals for the network capacity (i.e., the sum of the throughputs of individual flows) versus the number of flows. These results are obtained repeating each test with twenty different combinations of selected flows. Note that flow patterns are chosen in such a way that each node can be at most the source/destination of one flow. The plot indicates that Hybrid significantly outperforms OLSR in all the considered scenarios, with throughput gains ranging from 25% to 55%. On the other hand, Hybrid provides smaller improvements with respect to PacketOPP for low numbers of flows (up to 18%), and attains comparable performance for higher numbers of flows. It is important to note that the primary objective of our solution is not to define a new routing protocol, alternative to OLSR and PacketOPP, but to show how existing routing paradigms can be adaptively combined to optimize the network performance in a widest range of scenarios. In fact, a specific routing protocol can achieve better performance than another routing protocol in one scenario, but worse in another one. Our routing framework is capable of discovering the best routing option for every scenario using a minimal knowledge of the network behaviour and limited overheads. However, the combination of two different routing protocols allows some flows to obtain a throughput value higher than the one achieved using a single routing strategy. This also leads to an increase in the network capacity in most of the cases.

Impact of shadowing intensity

We investigate the impact of the shadowing intensity on the routing performance. More precisely, we keep constant the path loss exponent (equal to 2) and the shadowing deviation (equal to 4), but we vary the

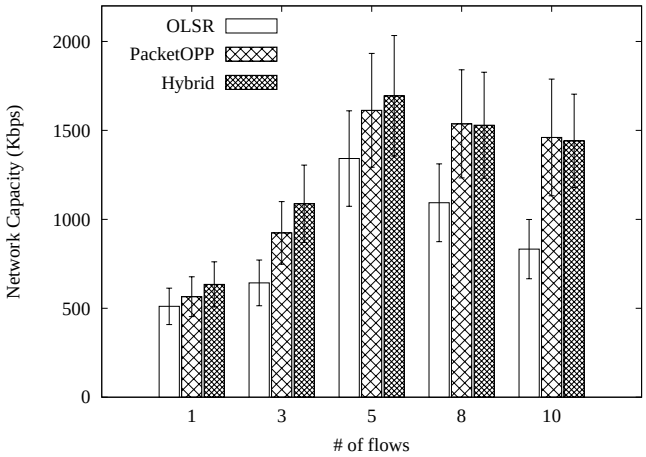
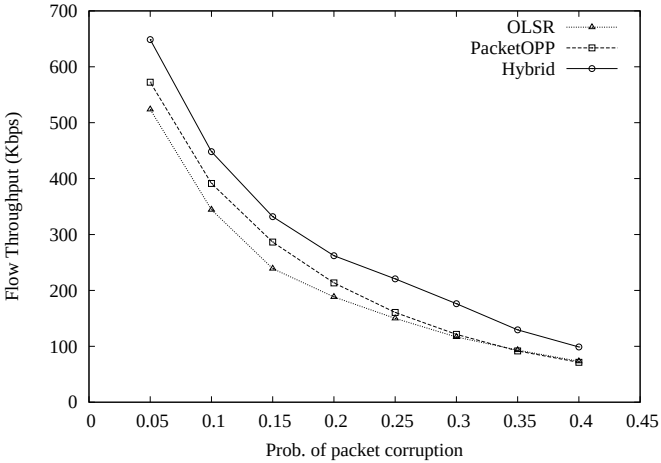


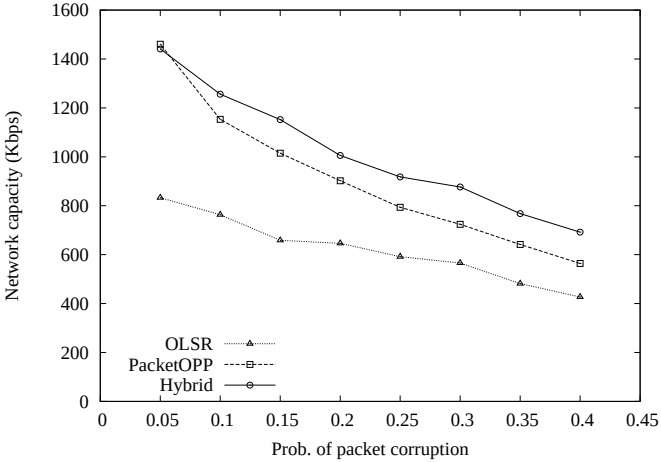
Figure 28: Comparison of network capacity for different touring protocols with varying number of simultaneous flows.

receiver sensitivity threshold to change the probability of packet distortion at the distance of 100m. Figure 29(a) shows the average throughput obtained in the case of a single flow, while Figure 29(b) shows the network capacity with ten randomly-selected simultaneous flows[8]. We do not report results for packet-corruption probabilities higher than 0.4 because the most of the links are too unreliable to allow OLSR to discover traditional shortest path between all node pairs. The results confirm that Hybrid efficiency is not affected by the channel unreliability. It is also interesting to observe that Hybrid has similar performance than PacketOPP when the probability of packet corruption is low, while it provides up to 20% throughput gains in more challenged environments. Moreover, unicast routing provides the worst network capacity in all the considered scenarios.

[8]The flow sets are the same as the ones used to obtain the results shown for the previous scenarios.



(a) One flow



(b) Ten flows

Figure 29: Comparison of throughput performance for different routing protocols with varying shadowing intensity.

5.5 Related Work

Due to the peculiarity of the proposed hybrid routing paradigm, we overview some representative protocols for ad-hoc networks based on configurable routing or reinforcement learning methods. Whereas Section 2.3 extensively survey wireless diversity-based solutions for WMNs, this Section is intended to clarify the state-of-the-art protocols more related to the hybrid architecture presented in this Chapter.

5.5.1 Configurable routing

The idea of developing a configurable routing framework to simultaneously support multiple ad hoc routing protocols is not new. Much work is dedicated to the development of component-based frameworks, such as Click (MKJFK00) and MANETKit (RC10) and many others, to enable the composition of basic routing functionalities (like packet classification, queueing, scheduling, and interfacing with network devices), to support dynamic routing reconfiguration, as well as to seamlessly integrate routing with middleware services. However, the focus of these platforms is to provide abstract programming interfaces, rather than QoS enhancements. Other studies are more oriented in providing customized routing solutions capable of supporting different QoS requirements and application scenarios. For instance, a middleware toolkit is proposed in (SP09) to implement dynamic path computation based on any combination of a number of supported QoS metrics. However, the focus is still on the software architecture needed for routing configuration rather than on routing performance optimization.

5.5.2 RL-based routing

Machine learning has gained much attention in recent years for solving specific routing problems in computer networks (For07). One of the first examples is the work by Boyan et al. (BL94) which defines a simple adaptive routing algorithm for static networks based on Q-Learning algorithm. More related to the field of ad hoc networks, is the paper (CHK04),

where reinforcement learning methods are used to control both packet routing decisions and node mobility, in order to improve the connectivity of the mobile network. More recently, Forster *et al.* in (FM11) have designed a multicast routing protocol for wireless sensor networks, which uses reinforcement learning to adaptively discover optimal routes with desired characteristics (e.g., route length, battery levels, etc.). A similar approach is adopted in paper (AVARGCCS07), which applies reinforcement learning and a Bayesian decision models to geographic routing in WSNs. These studies confirm that machine learning techniques are useful to improve routing performance in uncertain and unreliable environments. However, the use of reinforcement learning methods for implementing optimal routing composition has not received enough attention.

Chapter 6

Conclusions

In this thesis, we have deeply analyzed and discussed novel routing paradigms for wireless mesh networks. We have firstly presented an extensive overview of recent routing protocols designed to mitigate network performance degradation due to the unpredictability and unreliability of channel conditions. These solutions take advantage of the multiple recipients of a packet generated by a wireless transmission (wireless diversity) to face the inherent high packet loss rate, hence we have called this general strategy *wireless diversity-based*. We have identified two basic routing categories in this context: opportunistic routing and coding-based routing. The former generates multiple paths for each source-destination pair in a hop-by-hop fashion, and exploits this form of redundancy to provide resilience against path failures. The latter combines data packets to increase transmission robustness through data redundancy.

Keeping in mind the considerations mentioned above, we have focused our work on the opportunistic routing category, stimulated by the encouraging results obtained so far and motivated by the limitations of existing solutions. In principle, this routing paradigm constructs the routes incrementally, by selecting the best forwarder(s) at each hop according to channel conditions. However, one of the most relevant limitations of existing solutions is the selection of a set of the admissible

forwarders (or paths) before packet reception. Although this scheme contributes to keep coordination overhead limited, it significantly reduces the forwarding opportunities generated by a wireless transmission. Hence, we have presented and evaluated MaxOPP, a flexible and adaptive opportunistic routing algorithm able to select at each hop, and at run-time, the candidate forwarders that can maximize the opportunistic throughput gain. Forwarding decisions in MaxOPP are dynamically adapted to the variations of network conditions, ensuring an efficient trade-off between reliability, data redundancy and opportunistic gain. MaxOPP outperforms the shortest path routing with an average improvement that ranges from 10% with 4 flows to more that 100% with one flow in a grid-based topology, and arrives to 200% in a linear topology. MaxOPP is based on a randomized forwarding process to better exploit path diversity, but it may suffer from performance degradations when the number of flows increases. For this reason, we have proposed PacketOPP, a novel opportunistic routing protocol that wisely combine randomized opportunistic transmissions with packet scheduling to efficiently support multiple simultaneous flows. PacketOPP outperforms not only shortest path routing, but also MaxOPP and ROMER, another representative opportunistic routing protocol, with throughput gain varying from 70% for OLSR, 50% for ROMER and 10% for MaxOPP depending on the traffic scenario and the congestion level, whereas the fairness improvement is even higher, especially when there are many simultaneous flows.

Despite the flexibility provided by the hop-by-hop path construction, a higher degree of adaptability to network conditions is required in wireless mesh network environments. Link quality is highly variable both in space and time, due to wireless links peculiarities and characteristics of mesh application scenarios. Routing decisions are usually based on end-to-end principles, which do not take into account the localized channel dynamics. To face this additional issue, we have proposed RELADO, an opportunistic routing protocol that exploits the localized context to implement a more accurate selection of the possible forwarders after each packet transmission. RELADO combines end-to-end with localized in-

formation to dynamically selects the number of eligible forwarders for each packet transmission, so as to ensure transmission resilience across the network. With this flexibility, RELADO is able to reduce packet loss by providing the best trade-off between throughput maximization and packet progress. The throughput improvement provided by RELADO is about 46% over OLSR and about 30% over MaxOPP in scenarios with stable conditions. In general, compared to the other routing schemes, RELADO is able to boost throughput to higher values in both static and dynamic situations. Hence, this approach is particularly suitable whenever network conditions are critical, such as in case of node failures or abrupt degradations of link qualities.

There are many possible application scenarios related to WMNs. Thus, it is very difficult to design a general routing solution fitting the QoS needs of different applications. To conclude this thesis, we have proposed and evaluated a self-adaptive routing framework for WMNs, which enables the dynamic and on-the-fly combination of multiple routing strategies to maximize routing performance under arbitrary target objectives. The proposed framework relies on efficient machine learning techniques, which permits to tackle the complexity of the problem optimization without requiring to know how system performance depend on routing strategies and network conditions. Hybrid is able to outperform OLSR by more than 50% and PacketOPP about 20%, depending on the traffic characteristics. Moreover, the results confirm that Hybrid efficiency is not affected by the channel unreliability, since it can promptly react to channel variations adapting the routing strategy to the current network conditions.

The purpose of this thesis is to significantly contribute to the investigation of novel routing paradigms for WMNs, focusing on benefits and drawbacks of opportunistic routing. Hence, it provides useful insights on the capability of wireless diversity-based routing in improving WMNs performance. Moreover, it presents novel routing algorithms able to maximize throughput performance by employing flexible routing strategies able to adapt to network conditions and specific application requirements. All the presented results can be considered as a guideline for

the definition of innovative routing protocols for WMNs.

The promising results presented in this thesis encourage further work in several directions. One limitation of most of the existing diversity-based routing solutions for WMNs, is to consider throughput maximization as the main objective. However, this networking paradigm is expected to provide advanced communication services in diverse application scenarios, where routing must be efficient not only for bulk data transfers but also for real-time applications, which require stronger guarantees in terms of maximum delay and delay variability. Hence, we plan to design a routing approach able to adapt forwarding decisions not only to network and channel conditions, but also to QoS requirements of different applications.

An interesting property of a WMN is that the traffic flowing across the network can be distinguished between intra-mesh and extra-mesh. The former identifies traffic headed to nodes within the network, while the latter represents traffic directed to destinations belonging to external networks. We are interested in the development of a routing strategy able to serve intra-mesh and extra-mesh traffic with different policies, so as to achieve high performance by adapting forwarding decisions to traffic characteristics. As an example, we can consider that packets directed to the Internet have potentially multiple destinations within the mesh, since they can potentially reach their intended destination through any mesh gateway. Hence, extra-mesh traffic forwarding may be performed in a way that guarantees load balancing not only across the network but also among gateways, so as to avoid bottlenecks due to overloading of some gateways and underutilization of others. In addition, many common application scenarios, such as community networks, rely on heterogeneous Internet gateways (e.g. subscriber broadband access lines and high-speed provider fixed broadband links), which offer significantly different resources. For this purpose, we are planning to integrate gateway selection into the forwarding procedure, allowing either the source node or the forwarders to select the most convenient gateway(s) for each packet. This choice must be guided by the evaluation of the resources available at each specific gateway, combining information such as avail-

able bandwidth, traffic load and cost to reach this gateway.

To conclude, several open issues and specific application requirements motivate our interest in the development of novel routing paradigms for WMNs, able to ensure efficient and resilient communication in a more heterogeneous and large set of application environments.

References

[ABB+04] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris. Link-level Measurements from an 802.11b Mesh Network. *SIGCOMM Comput. Commun. Rev.*, 34(4):121–132, 2004. 3, 12, 89, 115

[ABC08] E. Ancillotti, R. Bruno, and M. Conti. Experimentation and Performance Evaluation of Rate Adaptation Algorithms in Wireless Mesh Networks. In *Proc. of ACM PE-WASUN'08*, pages 7–14, October 27 2008. 38

[ACLY00] R. Ahlswede, N. Cai, S.Y.R. Li, and R.W. Yeung. Network Information Flow. *IEEE/ACM Trans. on Inform. Theory*, 46(4):1204–1216, 2000. 22

[AVARGCCS07] R. Arroyo-Valles, R. Alaiz-Rodriguez, A. Guerrero-Curieses, and J. Cid-Sueiro. Q-Probabilistic Routing in Wireless Sensor Networks. In *IEEE ISSNIP'07*, December 3–6 2007. 125

[AWW05] I.F. Akyildiz, X. Wang, and W. Wang. Wireless mesh networks: a survey. *Computer Networks*, 47(4):445–487, March 2005. 2, 8, 10

[BCG05] R. Bruno, M. Conti, and E. Gregori. Mesh Networks: Commodity Multihop Ad Hoc Networks. *IEEE Communications Magazine*, 43(3):123–131, March 2005. 9, 10

[BGLAO09] J. Boice, J.J. Garcia-Luna-Aceves, and K. Obraczka. Combining on-demand and opportunistic routing for intermittently connected networks. *Ad Hoc Networks*, 7:201–218, 2009. 99

[BL94] J.L. Boyan and M.L. Littman. Packet Routing in Dynamically Changing Networks: A Reinforcement Learning Approach. *Advances in Neural Information Processing Systems*, 6, 1994. 124

[BM05] S. Biswas and R. Morris. ExOR: opportunistic multi-hop routing for wireless networks. *SIGCOMM Comput. Commun. Rev.*, 35(4):133–144, October 2005. 3, 4, 13, 14, 15, 16, 19, 20, 32, 33, 34, 45, 53, 59, 80, 113, 115

[CCL03] I. Chlamtac, M. Conti, and J.J.-N. Liu. Mobile ad hoc networking: imperatives and challenges. *Ad Hoc Networks Journal*, 1(1):13–64, July 2003. 12

[CG04a] M. Conti and S. Giordano. Multihop Ad Hoc Networking: The Reality. *IEEE Communications Magazine*, 45(4):88–95, April 2004. 1

[CG04b] M. Conti and S.. Giordano. Multihop Ad Hoc Networking: The Theory. *IEEE Communications Magazine*, 45(4):78–86, April 2004. 1

[CHK04] Y.-H. Chang, T. Ho, and L.P. Kaelbling. Mobilized ad-hoc networks: A reinforcement learning approach. In *Autonomic Computing*, pages 240–247, 2004. 124

[CJ03] T. Clausen and P. Jaquet. Optimized Link State Routing Protocol (OLSR). RFC 3626, October 2003. 3, 12, 72, 89, 115

[CJKK07] S. Chachulski, M. Jennings, S. Katti, and D. Katabi. Trading Structure for Randomness in Wireless Opportunistic Routing. In *Proc. of ACM SIGCOMM'07*, pages 169–180, August 27–31 2007. 3, 4, 13, 15, 29, 45, 46, 48

[CRSK06] J. Camp, J. Robinson, C. Steger, and E. Knightly. Measurement Driven Deployment of a Two-Tier Urban Mesh Access Network. In *Proc. of ACM MobiSys'06*, pages 96–109, Uppsala, Sweden, June, 19–22 2006. 3, 12, 80

[DADSC04] S.N. Diggavi, N. Al-Dhahir, A. Stamoulis, and A.R. Calderbank. Great expectations: the value of spatial diversity in wireless networks. *Proceedings of the IEEE*, 92(2):219–270, February 2004. 3, 14, 15

[DCABM03] D.S.J. De Couto, D. Aguayo, J. Bicket, and R. Morris. A High-Throughput Path Metric for Multi-Hop Wireless Routing. In *Proc. of ACM MobiCom'03*, pages 134–146, September, 14–19 2003. ix, 13, 32, 56, 59, 72

[DFGV07] H. Dubois-Ferriere, M. Grossglauser, and M. Vetterli. Least-Cost Opportunistic Routing. In *Proc. of Allerton Conference*, pages 994–1001, September 26-28 2007. 3, 19

[DPZ04] R. Draves, J. Padhye, and B. Zill. Routing in Multi-Radio, Multi-Hop Wireless Mesh Networks. In *Proc. of ACM MobiCom'04*, pages 114–128, Sept. 26–Oct. 1 2004. 13

[FJ93] S. Floyd and V. Jacobson. Random Early Detection gateways for Congestion Avoidance. *IEEE/ACM Trans. Netw.*, 1(4):397–413, August 1993. 69

[FKM+07] C. Fragouli, D. Katabi, A. Markopoulou, M. Médard, and H. Rahul. Wireless Network Coding: Opportunities & Challenges. In *Proc. of MILCOM 2007*, pages 1–8, October 29–31 2007. 22

[FM11] A. Forster and A. Murphy. Froms: A Failure Tolerant and Mobility Enabled Multicast Routing Paradigm with Reinforcement Learning for WSNs. *Ad Hoc Networks*, 9(5):940–955, July 2011. 125

[For07] A. Forster. Machine Learning Techniques Applied to Wireless Ad-Hoc Networks: Guide and Survey. In *IEEE ISSNIP'07*, December 3–6 2007. 99, 124

[GHK+07] C. Gkantsidis, W. Hu, P. Key, B. Radunovic, P. Rodriguez, and S. Gheorghiu. Multipath Code Casting for Wireless Mesh Networks. In *Proc. of ACM CoNEXT'07*, pages 1–12, December 10–13 2007. 24, 43

[GK04] Y. Ganjali and A. Keshavarzian. Load balancing in ad hoc networks: single-path routing vs. multi-path routing. In *Proc. of IEEE INFOCOM'04*, volume 2, pages 1120–1125, march 7–11 2004. 13

[GS08] M. Genetzakis and V.A. Siris. A Contention-Aware Routing Metric for Multi-Rate Multi-Radio Mesh Networks. In *Proc. of IEEE SECON 2008*, pages 242–250, June 16–20 2008. 13

[HMS+03] T. Ho, M. Médard, J. Shi, M. Effros, and D.R. Karger. On Randomized Network Coding. In *Proc. of 41st Allerton Conf.*, pages 11–20, October 1–3 2003. 22, 25

[Jai84] Jain, R., and Chiu, D., and Hawe, W. A quantitative measure of fairness and discrimination for resource allocation in shared computer systems. http://www.cse.wustl.edu/ jain/papers/fairness.htm, September 1984. 75

[JHM07] D. Johnson, Y. Hu, and D. Maltz. The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4. RFC 4728, February 2007. 3, 12

[KHW08] D. Koutsonikolas, Y. C. Hu, and C.-C. Wang. XCOR: Synergistic Interflow Network Coding and Opportunistic Routing. In *Proc. of ACM MobiCom'08 SRC*, September 14–19 2008. 30, 48

[KRH+08] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft. XORs in the air: practical wireless network coding. *IEEE/ACM Trans. Netw.*, 16(3):497–510, 2008. 3, 13, 14, 24, 26, 39, 41, 48

[LMK05] D.S. Lun, M. Médard, and R. Koetter. Efficient Operation of Wireless Packet Networks Using Network Coding. In *Proc. of IWCT 2005*, June 6–10 2005. 25

[LW06] M. Lu and J. Wu. Social Welfare Based Routing in Ad hoc Networks. In *Proc. of ICPP'06*, pages 211–218, August 14–18 2006. 37

[LY03] S.Y.R. Li and N. Yeung, R.W.and Cai. Linear Network Coding. *IEEE/ACM Trans. on Inform. Theory*, 49(2):371–381, February 2003. 22, 25

[LZL08] Y. Lin, X. Zhang, and B. Li. CodeOR: Opportunistic routing in wireless mesh networks with segmented network coding. In *Proc. of IEEE ICNP'08*, pages 1092–1648, October 19–22 2008. 47, 48

[MBLD07] V. Mhatre, F. Baccelli, H. Lundgren, and C. Diot. Joint MAC-aware routing and load balancing in mesh networks. In *Proc. of ACM CoNEXT'07*, pages 1–12, December 10–13 2007. 13

[MGLA05] M. Mosko and Garcia-Luna-Aceves. Multipath Routing in Wireless Mesh Networks. In *Proc. of IEEE WiMesh'05*, September 28 2005. 13

[MKJFK00] R. Morris, E. Kohler, J. Jannotti, and M. Frans Kaashoek. The Click Modular Router. *ACM Transactions on Computer Systems*, 18(3), 2000. 124

[NSZN06] B. Ni, N. Santhapuri, Z. Zhong, and S. Nelakuditi. Routing with Opportunistically Coded Exchanges in Wireless Mesh Networks. In *Proc. of IEEE WiMesh 2006*, pages 157–159, September 25 2006. 26, 41, 50

[PBRD03] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc On-Demand Distance Vector (AODV) Routing. RFC 3561, July 2003. 3, 12, 99

[PG93] A.K. Parekh and R.G. Gallager. A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single Node Case. *IEEE/ACM Trans. Netw.*, 1(3):344–357, June 1993. 69

[PPC04] L. Pelusi, A. Passarella, and M. Conti. Opportunistic networking: data forwarding in disconnected mobile ad hoc networks. *IEEE Communications Magazine*, 44(11):134–141, November 2004. 14

[QB03] X. Qin and R. Berry. Exploiting multiuser diversity for medium access control in wireless networks. In *Proc. of IEEE INFOCOM'03*, volume 2, pages 1084–1094, March 30–April 3 2003. 14

[QXG+08] C. Qin, Y. Xian, C. Gray, N. Santhapuri, and S. Nelakuditi. i^2mix: Integration of intra-flow and inter-flow wireless network coding, 2008. 42, 44

[Rap02] T. Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall, 2002. 12

[RC10] R. Ramdhany and G. Coulson. MANETkit: A Framework for MANET Routing Protocols. *Ad Hoc & Sensor Wireless Networks*, pages 1–16, 2010. 124

[RSBA07] K. Ramachandran, I. Sheriff, E. Belding, and K. Almeroth. Routing Stability in Static Wireless Mesh Networks. In *Proc. of PAM 2007, LNCS 4427*, pages 73–82, Louvain-la-neuve, Belgium, April, 2007. 3, 63, 73

[RSMQ06] E. Rozner, J. Seshadri, Y. Mebta, and L. Qiu. Simple opportunistic routing protocol for wireless mesh networks. In *Proc. of IEEE WiMesh 2006*, pages 48–54, Sept. 25 2006. 4, 19, 20, 21, 34, 35, 39, 71

[RSMQ09] E. Rozner, J. Seshadri, Y. Mebta, and L. Qiu. SOAR: Simple Opportunistic Adaptive Routing Protocol for Wireless Mesh Networks. *IEEE/ACM Trans. on Mobile Comp..*, 8(12):1622–1634, Dec. 25 2009. 3, 53, 59, 60, 70

[RSW+08] S. Rayanchu, S. Sen, J Wu, S. Banerjee, and S. Sengupta. Loss-aware network coding for unicast wireless sessions: design, implementation, and performance evaluation. *SIGMETRICS Perform. Eval. Rev.*, 36(1):85–96, 2008. 24, 26, 42

[SB98] R.S. Sutton and A.G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, March 1998. 99, 100, 101, 102, 111

[SP09] N. Shillingford and C. Poellabauer. Configurable routing in mesh networks. In *IEEE HotMESH'09*, 2009. 124

[SRB07] S. Sengupta, S. Rayanchu, and S. Banerjee. An Analysis of Wireless Network Coding for Unicast Sessions: The Case for Coding-Aware Routing. In *Proc. of INFOCOM'07*, pages 1028–1036, May 6–12 2007. 41

[TE92] L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and schedulingpolicies for maximum throughput in multihop radio networks. *IEEE Trans. on Automatic Control*, 37(12):1936–1948, December 1992. 44

[The09] The Network Simulator NS-2. http://www.isi.edu/nsnam/ns/, 2009. 72

[W. 09] W. Cordeiro, ETX-OLSR module for NS-2. http://www.inf.ufrgs.br/ wlccordeiro/resources.html, 2009. 61

[WD92] C.J. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8:279–292, 1992. 100, 103

[WLL08] J. Wu, M. Lu, and F. Li. Utility-Based Opportunistic Routing in Multi-Hop Wireless Networks. In *Proc. of ICDCS'08*, pages 470–477, June 17–20 2008. 3, 36, 37, 80

[YWK05] Y. Yang, J. Wang, and R.H. Kravets. Designing routing metrics for mesh networks. In *Proc. of IEEE WiMesh*, Santa Clara, CA, USA, September, 26 2005. IEEE Press. 56

[YWK06] Y Yang, J. Wang, and R.. Kravets. Load-balanced Routing For Mesh Networks. *ACM Mobile Comp. and Comm. Review (M2CR)*, 1, 2006. 13

[YYW+05] Y. Yuan, H. Yang, S. Wong, S. Lu, and W. Arbaugh. ROMER: Resilient Opportunistic Mesh Routing for Wireless Mesh Networks. In *Proc. of IEEE WiMesh'05*, September 28 2005. 3, 20, 21, 37, 53, 59, 70, 72

[ZJ09] J. Zhang and X. Jia. Capacity analysis of wireless mesh networks with omni or directional antennas. In *IEEE INFOCOM'09*, 2009. 99

[ZKR07] A. Zubow, M. Kurth, and J.-P. Redlich. Multi-Channel Opportunistic Routing. In *Proc. of European Wireless 2007*, April 1–4 2007. 3, 35

[ZKR08] A. Zubow, M. Kurth, and J.-P. Redlich. Cooperative Opportunistic Routing Using Transmit Diversity in Wireless Mesh Networks. In *Proc. of IEEE INFOCOM'08*, pages 1310–1318, April 13–18 2008. 3, 38, 54

[ZL08a] X. Zhang and B. Li. Dice: a Game Theoretic Framework for Wireless Multipath Network Coding. In *Proc. of ACM MobiHoc'08*, pages 293–302, May 27–30 2008. 46

[ZL08b] X. Zhang and B. Li. Optimized Multipath Network Coding in Lossy Wireless Networks. In *Proc. of IEEE ICDCS'08*, pages 243–250, June 17–20 2008. 46

[ZLZ08] K. Zeng, W. Lou, and H. Zhai. Capacity of opportunistic routing in multi-rate and multi-hop wireless networks. *IEEE Transactions on Wireless Communications*, 7(12):1–11, December 2008. 99

[ZWNL06] Z. Zhong, J. Wang, S. Nelakuditi, and G.-H. Lu. On selection of candidates for opportunistic anypath forwarding. *SIGMETRICS Mob. Comput. Commun. Rev.*, 10(4):1–2, October 2006. 19

[ZWR08] P. Zou, X. Wang, and R. Rao. Asymptotic Capacity of Infrastructure Wireless Mesh Networks. *IEEE Trans. Mobile Comput.*, 7(8):1011–1024, August 2008. 99

Unless otherwise expressly stated, all original material of whatever nature created by Maddalena Nurchis and included in this thesis, is licensed under a Creative Commons Attribution Noncommercial Share Alike 2.5 Italy License.

Check creativecommons.org/licenses/by-nc-sa/2.5/it/ for the legal code of the full license.

Ask the author about other uses.